Formal Description of Geometrical Properties

Tuan Minh Pham

UNIVERSITÉ DE NICE - SOPHIA ANTIPOLIS

ÉCOLE DOCTORALE STIC

SCIENCES ET TECHNOLOGIES DE L'INFORMATION ET DE LA COMMUNICATION

T H E S E

pour obtenir le titre de

Docteur en Sciences

de l'Université de Nice - Sophia Antipolis

Spécialité : Informatique

Présentée et soutenue par

Tuan-Minh PHAM

Description formelle de propriétés géométriques

Thèse dirigée par Yves BERTOT

et préparée à l'INRIA Sophia Antipolis, projet Marelle

soutenue le 21 Novembre 2011

Jury :

Jean-François Dufourd	-	Université de Strasbourg (Rapporteur)
Julio Rubio	-	Université de La Rioja (Rapporteur)
Jacques Fleuriot	-	Université de Édimbourg (Examinateur)
Julien Narboux	-	Université de Strasbourg (Examinateur)
Yves Bertot	-	INRIA-Sophia Antipolis (Directeur)

Résumé

La géométrie synthétique, aussi parfois appelée géométrie axiomatique, s'intéresse purement aux objets géométriques. En reposant sur des axiomes et des théorèmes qui mettent en relation les concepts de base de la géométrie, elle permet de mettre en valeur les propriétés géométriques pendant les preuves mathématiques.

Cette thèse se concentre sur les propriétés géométrique et leur description dans le système de preuve Coq. Les deux résultats principaux sont une bibliothèque de descriptions formelles et une extension du système de preuve pour permettre une interaction directe avec les objets géométriques pendant les preuves.

La première partie présente notre formalisation de la géométrie euclidienne basée sur la géométrie affine. Nous approchons les notions, les propriétés, et les théorèmes dans un style similaire à celui utilisé dans l'enseignement au lycée. Notre développement améliore le développement fourni précédemment par F. Guilhot en éliminant les axiomes inutiles, en fournissant des définition mieux appropriées pour certains objets géométriques, et en reformalisant leurs propriétés.

La deuxième partie s'intéresse à la question de l'orientation dans le plan. Cela permet d'enlever certaines ambiguïtés dans la présentation des objets géométriques et d'énoncer des problèmes de géométrie "ordonnée" et de les prouver.

La troisième partie s'intéresse à des questions de fondement. En particulier, nous montrons que les systèmes d'axiomes de Hilbert et Tarski peuvent être modélisés dans notre système. Nous montrons également que notre système d'axiome peut supporter les outils de preuve automatique basés sur la méthode des aires ou sur les bases de Gröbner. Le travail sur les bases de Gröbner a été effectué en collaboration avec J. Narboux.

La quatrième partie présente une combinaison de l'outil de preuve formel Coq avec l'outil de géométrie dynamique GeoGebra, en se reposant sur l'interface d'utilisation pour Coq développée en Java et appelée Pcoq. Le résultat de cette combinaison permet aux utilisateurs d'effectuer facilement des raisonnements géométriques dans le style de la géométrie du lycée, interactivement et avec le support d'une interface graphique. Ceci montre comment un système de preuve pourrait être utilisé en éducation.

Mots-clefs: Formalisation de la géométrie, Preuve interactive, Coq, logiciel de géométrie dynamique.

Abstract

Synthetic geometry, sometimes also called axiomatic geometry, deals purely with geometric objects. Relying on axioms and theorems relating the basic concepts of geometry makes it possible to highlight the geometrical properties during the proofs.

This thesis concentrates on geometrical properties and their description in the Coq proof assistant. The two main results of this thesis are a library of formal descriptions and a proof system extension to interact directly with geometrical objects during proofs.

The first part presents our formalization of Euclidean geometry based on affine geometry. We approach notions, properties, and theorems in a style similar to what is taught in high school. This development improves on the library developed by F. Guilhot by eliminating needless axioms, giving more appropriate definitions for some geometric notions and re-formalizing their properties.

The second part deals with the notion of orientation in the plane. It allows us to remove ambiguities in the presentation of geometric objects and state and solve ordered geometry problems.

The third part deals with foundations. In particular, we show that the axiom systems of Hilbert and Tarski can be described on top of ours. We also show that automatic proof methods like the area method and Gröbner bases can be integrated. The work on the area method was performed jointly with J. Narboux.

The fourth part presents a combination of the Coq formal proof tool and the Geogebra dynamic geometry tool. This is based on the java-based user-interface for Coq named Pcoq. This combination allows users to easily perform geometry reasoning as taught in high school and in a interactive manner with the support of a graphic interface. This shows how a proof system could be used in education.

Key words: formalization of geometry, interactive proofs, Coq, dynamic geometry software

Contents

1 General Introduction **1**

1.1 Teaching and learning geometry 1

1.2 Using dynamic geometry software 3

1.3 Formal proof with a proof assistant 4

1.4 Our work 5

2 Formalization of Elementary Geometry **6**

2.1 Introduction 6

2.2 A formalization of high-school geometry 8

2.3 Reducing axioms and having a constructive library in a geometric view . 12

2.4 Formalization of affine geometry 13

2.4.1 Formalization of vector 16

2.4.2 Formalization of line 22

2.5 Formalization of Euclidean geometry 26

2.5.1 Cartesian coordinate system 28

2.5.2 Elementary constructions 31

2.5.3 Compound constructions 33

2.5.4 Angle and Trigonometry 34

2.5.5 Plane transformation 38

2.6 Discussion 39

2.7 An example about school proof and formal proof 41

2.7.1 The proof in school 42

2.7.2 The formal proof 42

2.7.3 Discussion 45

2.8 Conclusion and perspectives 46

3 Orientation and its applications **47**
3.1 Introduction . . . 47
3.2 Definition of orientation . . . 49
3.3 Some properties of orientation . . . 50
3.3.1 The properties of Knuth's system . . . 50
3.3.2 Proving the properties . . . 51
3.3.3 Variants . . . 56
3.4 Orientation and order . . . 56
3.4.1 Properties . . . 56
3.4.2 Proving the properties . . . 59
3.5 Some application . . . 61
3.5.1 Orientation in formalizing some geometric notions . . . 61
3.5.2 Orientation in Proving Theorems . . . 67
3.6 Some discussions about full angles . . . 70
3.7 Conclusion . . . 71

4 Multiple Points of View about Geometry **72**
4.1 Introduction . . . 72
4.2 Verifying axiomatic systems of synthetic geometry . . . 73
4.2.1 Verifying Hilbert's system . . . 73
4.2.2 Verifying Tarski's system . . . 81
4.3 Combining with the area method . . . 82
4.3.1 The area method . . . 82
4.3.2 Correctness of the area method . . . 85
4.3.3 Usability of the area method . . . 88
4.4 Combining with the method of Gröbner bases . . . 95
4.5 Conclusion and future work . . . 97

5 Dynamic Geometry Tool for Interactive Theorem Proving **99**
5.1 Context . . . 99
5.2 Introduction to Geogebra . . . 101
5.3 Some analyses about school proofs . . . 103
5.4 The interface with the proof related feature . . . 105
5.4.1 A snapshot of the interface . . . 105
5.4.2 Stating theorems . . . 106
5.4.3 Theorem proving . . . 108

5.5 Some discussion about proving with our interface . . . 114
5.6 Communication between Geogebra with Coq . . . 114
5.7 Implementation . . . 115
5.7.1 Integration of the prove view . . . 116
5.7.2 Tree structure for geometric object definition . . . 117
5.7.3 Interactions with users during their proofs . . . 118
5.7.4 Visualizing theorem statements . . . 120
5.8 Conclusion and Future Work . . . 120

6 General Conclusion and Perpectives **122**
6.1 Conclusion . . . 122
6.2 Perspective . . . 123

A Formal proof of a geometry theorem **126**

B Axiomatic system **129**

Chapter 1

General Introduction

1.1 Teaching and learning geometry

Geometry is an important branch of mathematics, that deals with properties, measurement and relationships between geometric objects. Learning geometry is a brain-boosting activity that helps improve student's brain function. Doing geometry proofs requires the brain to operate in new and complex ways. This requires logical thinking, a mental process that is rarely well developed in the younger students.

The conventional approach to geometry is synthetic or axiomatic geometry, which deals purely with geometric objects directly by geometrical properties. This is the kind of geometry, for which Euclid is famous, that makes use of axioms, theorems and logical arguments to draw conclusions. Its formal proofs are considered as sequences of geometric deductive reasoning from axioms. They are taught in high school with a well-known method called two-column method. A two-column geometric proof consists of a list of statements, and the reasons showing that those statements are true. The statements are listed in a column on the left, and the reasons for which the statements can be made are listed in the right column. Each reasoning step is a row in the two-column proof.

The use of formal proofs is always subject of debates for educators - see the survey of Battista & Clements [3]. Some argue that we should continue the traditional focus on axiomatic systems and proofs. Some believe that we should abandon proof for a less formal investigation of geometric ideas. Others believe that students should move gradually from an informal investigation of geometry to a more proof-oriented focus. High school geometry with its formal proofs is considered hard and very detached from practical life for some reasons: lack of proof and proving in earlier school years, lack of understanding of geometry concepts and student's cognitive development.

The research of Piaget on student's cognitive development shows that thinking in general progresses from being non-reflective and unsystematic, to empirical, and finally to logical-deductive. The research of Van Hiele on understanding of geometry concepts suggests that students' geometrical understanding progresses through various levels in Tab.1.1.

- Level 1 - Visualization : Students can name and recognize shapes by their appearance, but cannot specifically identify properties of shapes.
- Level 2 - Analysis : Students begin to identify properties of shapes and learn to use appropriate vocabulary related to properties, but do not make connections between different shapes and their properties.
- Level 3 - Argumentation : Students are able to recognize relationships between and among properties of shapes or classes of shapes and are able to follow logical arguments using such properties.
- Level 4 - Deduction : Students can go beyond just identifying characteristics of shapes and are able to construct proofs using postulates or axioms and definitions. A typical high school geometry course should be taught at this level.

Table 1.1: Students' geometrical understanding progresses through various levels which cannot be skipped

Both theories suggest that students must pass through lower levels of geometric thought to attain higher level and that they can understand and explicitly work with axiomatic systems only after they have reached the highest levels. Thus, the explicit study of axiomatic systems is unlikely to be productive for the vast majority of students in high school.

Moreover, Battista and Clements suggest that the most effective way to include meaningful proof in classes is to avoid formal proof at the beginning and focus instead on justifying ideas based on visual and empirical foundations. This can gradually lead students to appreciate the need for formal proof. In fact, learning by inductive reasoning alone is "surface learning" and deductive proof should continue to be an essential part of geometry curricula.

1.2 Using dynamic geometry software

Consistent with this alternative to axiomatic approaches, the focus of dynamic geometry softwares (DGSs) is to facilitate students' making and testing conjectures. Dynamic geometry softwares not only allow students to understand construction steps that lead to final drawings, but also provide access to geometric objects, allow students to move free points and observe the influence on the rest. This allows students to explore new implicit properties from the drawings. Some among the numerous dynamic geometry softwares provide a justification feature with the support of algebraic (coordinate-based) methods or semi-algebraic (coordinate-free) methods that facilitate justifying conjectures.

Nowadays, dynamic geometry softwares are widely used in school. Many researches pointed out the important role of dynamic geometry softwares in learning and teaching geometry. The use of dynamic geometry softwares can further students progress. The research on the influence of dynamic geometry softwares on students' progress with respect to van Hiele levels [15][21] showed that dynamic geometry softwares may help students pass through van Hiele levels. However, most of the material proposed for the classroom seems to only be concerned with the first three levels.

The fourth level deals with deductive proof. In fact, algebraic (coordinate-based) methods or semi-algebraic (coordinate-free) methods only allows to justify conjecture. They do not offers a capability to build deductive proofs. In particular, algebraic methods are able to produce non-trivial geometric proofs automatically, but their proofs are usually long and hard to read. On the other hand, coordinate-free methods only use geometrically meaningful quantities and their generated proofs are human-readable. But these proofs still are not purely traditional deductive proofs, not produced in the manner taught in school.

The exploration and proof activities should be interlaced. I think that these two activities could be better interlaced if they were both conducted using the computer. This leads to the necessity of developing a geometry proving tool for high school students which they can use to to construct geometry object, explore conjecture and interactively construct traditional geometry proofs. Proving with a such system enables students to understand geometric concepts deeply and beyond the use of these concepts in scope of current being studied geometry problems.

1.3 Formal proof with a proof assistant

A proof assistant or interactive theorem prover is a tool to develop formal proofs guided by users. Proof assistants allow us to state mathematical theorems, perform logic reasonings and they verify the validity of logical steps.

Proofs are developed in a interactive manner that requires logical reasoning capabilities from users. Therefore, using proof assistants could help students to better understand the concepts of deduction.

This use has more interest in proving geometry theorems because traditional geometry reasoning usually relies on tacit assumptions which are based on visual evidence. The lack of proving these conditions leads to uncertified proofs. It even leads to wrong reasonings when assumptions are obtained from incorrect drawing. Using a proof assistant has the following advantages:

- It gives a clear logical view about geometric problems. Students understand well what are hypotheses and conclusions. They understand well the logical inferences used in each reasoning step by observing the change of proof environment(including hypotheses and conclusions).
- Students can construct proofs step by step interactively. Reasoning steps are verified by the proof assistant, thus constructed proofs have very high level of confidence.
- It allows to combine purely geometric arguments with other kind of proofs.

However, beside the advantages, current proof assistants can not be used in high-school without being adapted for several reasons:

- The syntax is not adapted to high-school students.
- The level of detail required in the proof is too high. Even a simple proof requires proving many lemmas.
- The developed formal proofs are not readable and understandable

If we aim at using proof assistant in proving geometric theorems, these inconveniences need to be corrected.

1.4 Our work

We see in the last sections the need for *formal proofs* and the significance of *geometry dynamic softwares* in education as well as the use of *proof assistants* in developing formal proofs. Our work presented in this dissertation is an attempt to combine these notions. In particular, we approach interactive formal proofs of geometric theorems using the Coq proof assistant and the combination of these proofs with the Geogebra geometry dynamic software. Our work contains 4 principal part as follows:

To be able to develop proofs, we need to have a library containing the necessary notions, properties and basic theorems. Constructing this library is the first part of our work. At this moment, we focus only on high school geometry. The covered notions, properties, and basic theorems make it possible to construct geometric proofs by using logical reasoning as taught in high school. This work is partially presented in [44]

However, a main difference between proofs in high school and formal proofs is that the latter require a high level of detail. Every case has to be considered and every fact used in proofs is either an axiom or an theorem that is already proved before.

We have detected an area of geometry where implicit assumptions are often made, using only drawing as justification. This area is the area of ordered geometry, which deals with relative position of points along a straight line, around a polygon or a circle, orientation of the plane or convexity. Describing order geometry is the subject of the second part.

We formalize the notion of orientation that allows us to remove ambiguities in the presentation of geometric objects and state and solve ordered geometry problems. This work is partially presented in [42]

Our library is not limited to interactive proving. Automatic proving based on the the area method and Gröbner bases can be integrated it. This allows users to switch between different proof modes. These integrations are considered in the third part. This work is partially presented in [44]

The last part of our work aims at combining our library in Geogebra. The significance of our work is increased manyfold by this combination. We enrich it with an interactive proving feature that makes it possible to develop traditional geometric proofs by clicks of the mouse and interaction with geometrical objects during proofs. This proving feature offers a logical view for geometry problems, motivates student to verify the validity of their conjecture that they explore from figures. Interactive proving with the interaction of dynamic geometry is a good direction for educational application. This work is presented in [43].

Chapter 2

Formalization of Elementary Geometry

2.1 Introduction

The earliest axiomatic system for geometry was presented by Euclid in Euclid's Elements which has been one of the most influential books in the history of geometry. However, there were many discussions concerning the question of whether this system was fundamentally sound and consistent. In fact many flaws were found in proofs in this text. For instance, the *superposition* argument asserting that, by moving a triangle, the sides and the shape are preserved, is not a consequence of the axioms; axioms of order, etc. We refer the reader to [24][29] for more examples. These documents show that figures accompany many of the proofs. Using implicit assumptions from intuition and reasoning from diagrams make proofs incomplete and not rigorous. These gaps lead to the appearance of other axiomatic systems in an attempt to provide a formal axiomatic for Euclidean geometry, where tacit assumptions are made explicit.

Many such axiom systems have been proposed. First, we can cite the system that was proposed by Hilbert in The Foundations of Geometry [30]. This is regarded as a more formal version of Euclid's system. This system is based on 3 primitive notions: point, line, and plane. It contains 6 predicates and 20 axioms which are classified in five groups: incidence axioms, order axioms, congruence axioms, parallelism axioms and continuity axioms. It's purely geometrical and none of theses axioms concern numbers or arithmetic. This was not the first, but it is perhaps the most intuitive and the closest to Euclid's.

Some work was performed to formalize this system. A formalization in Coq was proposed by C. Dehlinger *et al.* [16] and another in Isabelle/Isar was proposed by

L. Meikle and J. Fleuriot [36]. These formalizations using proof assistants show that, although Hilbert tried to provide a formal system, proofs in his system are still not fully formal. In particular, degenerate cases are often implicit, geometric intuitions are still interleaved in the proofs for many theorems.

Another system for geometry was proposed by Tarski [47]. Its last version contains only 11 axioms, 2 predicates, and only one primitive notion: the notion of point. In comparison with Hilbert's, this system is simpler, it relies only on first-order logic whereas the last axiom group of Hilbert requires second-order logic. This system is also less intuitive, it involves only one intuitive notion of triangle and predicates about betweeness and congruence.

A formalization of Tarski's system was realized by J.Narboux in Coq [40]. This formalization indicates that, in comparison with Hilbert's system, the use of Tarski's system leads to more uniform proofs with less degenerate cases. Moreover, because this system is only based on points, it can easily be generalized to other dimensions by just changing the dimension axiom. This cannot easily be done in Hilbert's system.

In *Learning and Teaching Axiomatic Geometry* [49], M. Stone said that there is quite general agreement that in all of school mathematics, there is no subject more difficult to learn or to teach than axiomatic geometry. Indeed, students have widely different geometric insights that lead to quite different inferences, and in order to comprehend adequately axiomatic geometry, students have to have some knowledge and understanding of logical principles. Thus, with the aim of application in school curriculum, we need to formalize geometry from a moderately complex axiomatic system, that is pedagogically suitable and satisfactory from a logical point of view. However, Hilbert's or Tarski's systems have too low a starting point, hence they are too far for school use. Proofs are, at the level of axioms, far from being easy, hence very difficult for students.

We are interested in the development of F. Guilhot, a high-school teacher who developed a library in the Coq proof assistant for interactive theorem proving in geometry at the high-school level [27][28]. This development is based on a specific axiom system which is adapted to the knowledge of high-school students. It covers a large portion of the basic notions of plane geometry, the properties and the theorems found in the high-school geometry programme. Moreover, its proofs and its geometry reasoning are close to what students learn in high-school. Some classical theorems are proved in this library, such as Menelaus, Ceva, Desargues, Pythagoras, Simsons' line, etc.

However, the system was constructed with a pedagogical view rather than a logical view. This explains the existence of some defects for this library. The axioms are redundant and constructive definitions are missing. The fact that geometric objects are usually defined by axioms stating their properties is a usual approach in high-school curriculum. However, this leads to an explosion of the number of axioms with many redundant ones. The use of compound geometric constructions, such as, for example intersection point of lines, orthogonal projection of a point on a line etc, without knowing how they are constructed makes the library lose constructive property.

In spite of these drawbacks, the library is meaningful from a pedagogical perspective. The covered notions, properties, and theorems fit the high-school geometry curricula. The system allows us to interactively construct geometric reasoning. These motivate us to work on this library. We dedicate this chapter to present our work in this direction. First, we give an overview about the library, we detail its drawbacks and draw up strategies to improve it. We then present our development in Coq, focusing on plane geometry. Finally, we discuss the difficult points that we met in our formalization work and draw conclusions.

2.2 A formalization of high-school geometry

The initial idea of a library in Coq for high-school geometry was proposed by F. Guilhot - a highschool teacher [27][28]. With the aim of illustrating geometry proofs in high-school class room, she developed a library based on educative formation in high-school. Her development is organized as in Fig.2.1.

F. Guilhot started her formalization with constructing affine geometry. Algebraic structures and vector spaces are never addressed in school programme. However, the notions of mass point and barycenter are presented in high-school courses, and calculations of mass point and barycenter are straightforward and familiar to students. Therefore, F. Guilhot constructed affine space from mass point using the universal space proposed by M.Berger [4]. This construction is explained in [28].

To build up the Euclidean space, the affine structure is enriched with the notion of scalar product of vectors (denoted by $\overrightarrow{u} \cdot \overrightarrow{v}$) which makes it possible to define an Euclidean distance, a measure of angle, etc. Geometric notions are added one by one in the order that they are taught in school.

With about 25000 lines of formalization in 70 files, a large part of the French high-school geometry programme is covered, many geometric notions are formalized and

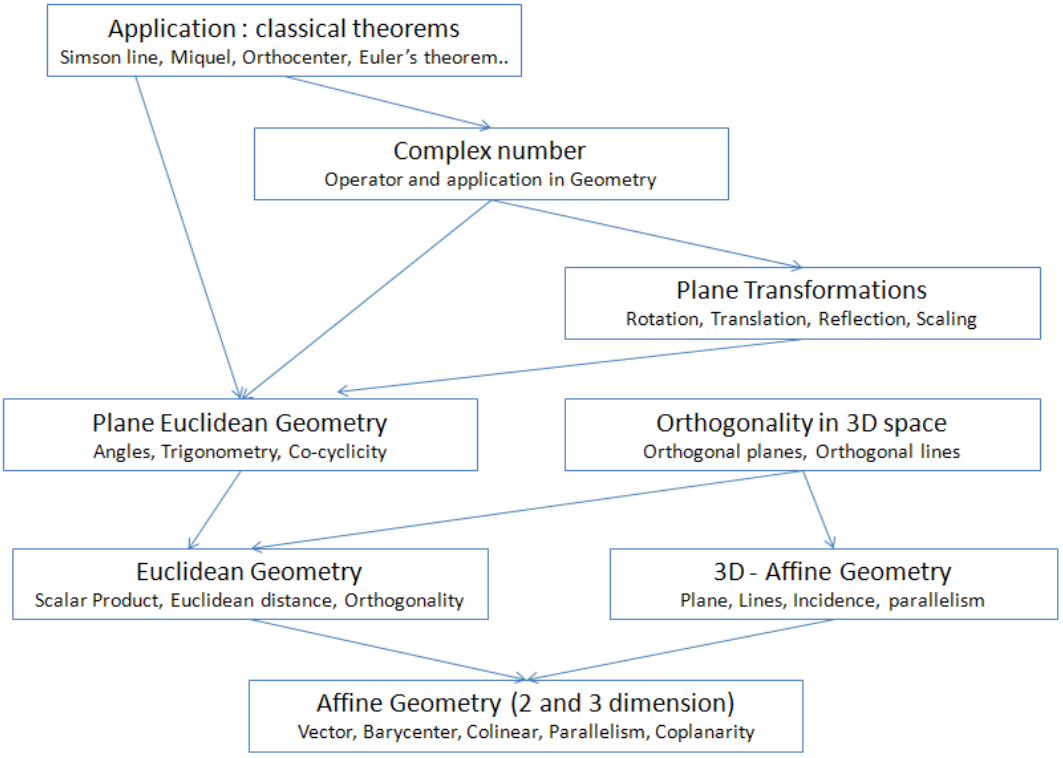


Figure 2.1: Structure of F. Guilhot's development

some classical theorems are proved. The detail of formalized geometric notions is as follows.

- General affine geometry : basic notions of affine geometry in any dimension are formalized
  - Point, mass point, vector, algebraic measure
  - Barycenter, midpoint, barycenter properties, gravity center of triangle
  - Parallel lines, concurrent lines
  - Collinearity, co-planarity
- Affine geometry of dimension 3 : properties of parallelism and incidence of lines and planes in 3D space are formalized
- General Euclidean geometry : the affine structure is enriched with scalar product and the following dependent notions
  - scalar product and its properties

  - orthogonality of vectors and lines
  - orthogonal projection of a point on a given line
  - Euclidean distance

- Orthogonality in dimension 3 : orthogonal lines and orthogonal planes are added
- Euclidean plane geometry : this is the most important part of high-school geometry. It includes
  - Notions defined using Euclidean distance : perpendicular bisectors, circles
  - Coordinates system, orthogonal coordinates system, Cartesian coordinates system
  - Oriented angle of vectors and lines
  - Trigonometry
  - Signed areas of triangles and parallelograms, determinant
- Plane transformation : study properties of plane transformation and composed transformations, particularly properties about preservation of length and angle measure

F. Guilhot does not try to provide a system with a minimal number of axioms, nor to provide an automated tool for theorem proving. She does not build up the whole of Euclidean geometry from a fundamental axiomatic system such as the systems of Hilbert or Tarski. An alternative approach is used to arrive at the same geometrical results, where definitions of geometric notions, theorems and geometry reasonings are described as they are taught in high-school. In particular, geometric notions usually are declared as abstract notions, each notion has some companion properties which are stated in the form of axioms to manipulate it. For example, the following is the formalization of the notion of line. An abstract type for lines and an abstract function to construct a line from two points are declared. Two axioms about line properties are introduced, making it possible to manipulate this notion.

```
Variable Line : Type.
Variable line : Point -> Point -> Line.
Axiom
  line_permute : forall A B : Point,
  A <> B -> line A B = line B A.
Axiom
  alignes_line :  forall A B C : Point,
    A <> B -> A <> C  -> collinear A B C -> line A B = line A C.
```

From the pedagogical point of view, defining by properties is a usual manner to define an object in high-school. Students approach geometric notions through properties. This way of defining is appropriate for notions considered to be really primitive notions. But axioms are unnecessary in defining compound notions, which are composed from other ones, and axioms are unnecessary for relations between geometric objects as well. The following formalization is to define orthogonal projection of a point on a line and parallelism relation of lines.

```
Variable orthogonalProjection : Point->Point->Point->Point.
Axiom def_orthogonalProjection1 :
 forall A B C H : Point,
 A <> B ->
 orthogonalProjection A B C H ->
 collinear A B H /\ orthogonal (vector A B) (vector H C).
Axiom def_orthogonalProjection2 :
 forall A B C H : Point,
 A <> B ->
 collinear A B H ->
 orthogonal (vector A B) (vector H C) ->
 orthogonalProjection A B C H.

Variable parallel : Line -> Line -> Prop.
Axiom def_parallel :
 forall (A B C D : Point) (k : R),
 A <> B -> C <> D ->
 1B + (-1)A = kD + (-k)C ->
 parallel (line A B) (line C D).
Axiom def_parallel2 :
 forall A B C D : Point,
 A <> B -> C <> D  ->
 parallel (line A B) (line C D) ->
 exists k : R, 1B + (-1)A = kD + (-k)C.
```

For orthogonal projections, two axioms are introduced, the first one gets properties from the notion, and the second one inversely gets the corresponding object from the given properties. The first axiom gives us collinearity of the triple of A,B and H, and orthogonality of two vectors $\overrightarrow{AB}$ and $\overrightarrow{HC}$ if we have that H is the orthogonal projection point of C on line AB. The second asserts that H is the orthogonal projection point of C on line AB if we have collinearity of A,B and H, and orthogonality of $\overrightarrow{AB}$ and $\overrightarrow{HC}$.

For the parallelism of lines, two axioms assert that the parallelism of line AB and CD is equivalent to the collinearity of $\overrightarrow{AB}$ and $\overrightarrow{CD}$. This collinearity is expressed by $\overrightarrow{AB} = k \times \overrightarrow{CD}$

This is a fast way to have a geometry library, which provides notions and properties for students to construct proofs of other theorems. However, this leads to an explosion of the number of axioms and primitive notions. They increase linearly in respect to the number of added notions, with a lot of redundancy. As a result of this, there are about 127 axioms in the library. In our examples, we can define the parallelism of 2 lines AB and CD by collinearity of 2 vectors $\overrightarrow{AB}$ and $\overrightarrow{CD}$. Therefore, the two axioms defining parallelism could be dispensed with.

Moreover, the library does not provide functions to construct points, lines and compound geometric constructions. They are used in axioms which state their properties without knowing how they are constructed. This makes the library lose the constructive property. This is the case of orthogonal projection in our example. In a constructive view, we can construct the orthogonal projection of C on line A B by getting the intersection point of line AB with the line which passes through C and is perpendicular with line AB. Besides, the lack of definitions in the form of functions leads to unusual representations. For instance, *orthogonalProjection* $A\ B\ C\ H$ expresses the relations of points $col\ ABH \wedge CH \perp AB$ rather than providing a definition of H.

2.3 Reducing axioms and having a constructive library in a geometric view

In spite of its drawbacks, this library is good from a pedagogical point of view. It matches requirements for a geometric library being used in school. This motivates us to improve this library. Our objective is to find a more compact axiom system, that allows us to rebuild all geometric notions and to prove their properties in this library. To do that, we analyze the library and draw up strategies to reduce its axiom system.

We can distinguish geometric notions into geometric constructions and geometric relations. Geometric constructions are geometric objects such as points, lines, circles, angles, altitudes, orthocenters, etc. Geometric relations express relations between these geometric objects, for example the parallelism of lines, the perpendicularity of lines, etc. These relations are essential for the library to manipulate geometric objects and state geometric problems.

For the first category, we continue to divide them into primitive constructions and compound constructions. Primitive constructions are elementary constructions based on ruler and compass. Compound constructions are considered as sequences of elementary constructions. To make our library match dynamic geometry systems, we refer to [31] for the elementary construction list.

As compound constructions are constructed step by step using elementary constructions, axioms related to them can be eliminated by retracing the combinations of elementary construction. These axioms become theorems that have to be proved. For each compound construction, there are always at least two companion theorems that correspond to axioms used to define this construction in the system of F. Guilhot showed in the example of the previous section. One of these axioms is to prove properties from the given notion, and the other is to get the corresponding notion from properties. The latter leads to proving the unique existence of a construction with the given properties. This is not always straightforward. The axioms concerning the orthogonal projection of a point C on a line AB in the previous section is an example. We have to prove that there is only a point H such that A, B, and H are collinear and $AB \perp CH$. This allows us to deduce that if H satisfies these properties then H is equal with the orthogonal projection of C on AB which have the same properties.

Reducing axioms related to elementary constructions is more difficult. These constructions are primitive in the geometric view, but they may be non-primitive in the logical view. For example, lines and circles are primitive constructions. However, we can consider a line as a set of points that are collinear with the 2 given points, a circle as a set of points that have the same distance with respect to the center. Defining notions of this kind depends on the axiom system and the primitive notions from which we build up geometry. Their properties are proved from their definitions. Thus, axioms for these notions can be eliminated.

Once geometric objects are formalized, we have their definition and properties. Intuitively, it is not difficult to define relations between geometric objects. Thus, reducing the number of axioms used to define these relations comes without much effort.

Besides, there are some notions that come from algebra such as the signed area of a triangle, the trigonometric functions, etc. These notions are usually expressed by algebraic equations, so these equations can help us in definitions.

2.4 Formalization of affine geometry

In this section as well as in some of the next ones about formalization in Coq, we cannot give all the details about our development. We rather focus on some interesting, crucial formalizations of each part. This clarifies the technique we used to eliminate axioms from the library of F. Guilhot.

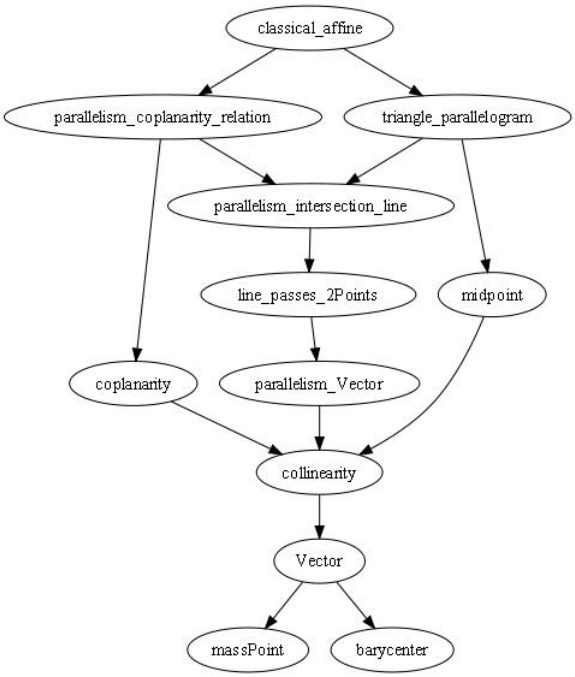


Figure 2.2: Formalization of Affine Geometry

Figure 2.2 shows the dependency between files in our formalization for affine geometry. The file, namely *mass point*, contains the formalization of F. Guilhot about the notion with the same name. The following axioms are used for mass point

```
Axiom Idempotency: nP + mP =(m+n)P.
Axiom Commutativity:  nP + mQ =  mQ + nP.
Axiom Associativity: nP + (mQ + kR) = (nP + mQ) + kR.
Axiom Definition of scalar multiplication: k(nP) = (k*n)P.
Axiom Distributivity k (nP + mQ) = knP + kmQ.
```

We use small letters a, b, c, etc. to denote real numbers; capital letters A, B, C, etc. to denote points; pairs consisting of a real number and a point in the form aA to denote mass points.

Here we work with the field of real numbers. It's familiar with high school students.

These axioms were not directly introduced in her formalization. Instead of this, she formalized mass point by making a mapping into an abstract field structure in Coq. Therefore, algebraic properties represented by these axioms is ensured. This formalization enables us to simplify equations of mass points using automated tactics from the Coq library for fields (such as `ring_simplify`, `field_simplify`, etc.). It makes calculations easier. The use of mass points is a good approach for computation, and familiar to students.

The file named *barycenter* is for the notion of barycenter. This notion allows us to create a new point R from the two given mass points nP and mQ in the case $n+m \neq 0$ such that $(n+m)R = nP + mQ$ and the new point R is called the barycenter of P and Q with the masses n and m respectively. The equality $(n+m)R = nP + mQ$ is rewritten by $(n+m)(barycenter\ np\ mQ) = nP + mQ$. The following axiom is used to define this notion.

```
Axiom Definition_of_barycenter : ∀ (m n:Real)(P Q :Point)
```
$m + n \neq 0 \rightarrow \exists(R : Point), nP + mQ = (m+n)R$

The notion of vector is considered as a sum of 2 mass points with the sum of their masses equals zero. This is represented as follows:

```
Definition vec (A B : Point) := (−1)A + 1B.
```

This definition allows us to convert vectors into mass points so that calculations of vector are translated to the ones of mass points, hence the automatic tactics can be reused. For the detail of this formalization, we refer the reader to publications by F. Guilhot[28][27]. However, we can simply understand that calculations of mass points and vectors are easily realized by some automatic tactics, as in the proofs of the following examples.

The first example states Chasles' relation for vector $\overrightarrow{AB} + \overrightarrow{BC} = \overrightarrow{AC}$.

```
Lemma test1_addVector: forall (A B C : Point),
vec A B + vec B C = vec A C.
Proof.
intros.
unfold vec in *.
RingMP.
Qed.
```

By unfolding the definition of vector in the goal, we have to prove that $((-1)A + 1B) + ((-1)B + 1C) = (-1)A + 1C$. The tactic *RingMP* allows us to solve this equality automatically. Simply speaking, the tactic repeatedly performs algebraic transformations to reduce 2 parts of the equality in a normalized form, so that their equality can be decided.

The next example is more complex, it states that if I is the midpoint of BC we have $\overrightarrow{AB} + \overrightarrow{AC} = 2\overrightarrow{AI}$. I is represented as the barycenter of B and C with masses 1 and 1 respectively.

```
Lemma calcul_midpoint: forall (A B C I :Point),
2I = 1B + 1C ->
vec A B + vec A C = 2 (vec A I) .
```

```
intros A B C I H.
unfold vec in *.
cut (1I = (/2) (1B +1C))(*assert this fact*).
intros H0.
rewrite H0 in *.(*replace 1I with ..*)
assert (2<>0).
intros H1.
FieldMP 2.
...
Qed.
```

The proof of this example is similar to the first example. By unfolding the definition of vector, we have to prove that $((-1)A + 1B) + ((-1)A + 1C) = 2((-1)A + 1I)$. Replacing $(1I)$ with $((/2)(1B + 1C))$ in the goal leads us to prove $((-1)A + 1B) + ((-1)A + 1C) = 2((-1)A + ((/2)(1B + 1C)))$. As in the first example, we have a tactic, namely *FieldMP*, to solve this equality. A difference between these tactics is that FieldMP is used for equality with fractions of R.

2.4.1 Formalization of vector

Until this point, we preserve the formalization of F. Guilhot. Let's consider the definition of vector. This notion plays a crucial role for our formalization, thus we give the details of formalizing this notion.

The definition of vector allows us to convert vectors into mass points so that calculations of vector are translated to the ones of mass points and can be automatically performed. However, it also has some drawbacks. It is a constructor of a vector from two points rather than a definition of vector. Vectors only are a special case of mass point combinations, they do not have a proper type. This leads to ambiguity in the type *MassPoint*. In particular, with a value v of type *MassPoint*, we do not know whether v is the representation of a vector or not. Furthermore, the lack of a clean type forces us to use vectors in the form of $\overrightarrow{AB}$, we do not have $\overrightarrow{v}$ without precise points. This leads to difficulties when defining geometric notions where vectors are used as arguments in the definition.

Data structure for vector

We aim at providing a new type for vector so that automatic calculations are inherited. In fact, a value of mass point v is a vector if there are 2 points A and B such that v can be expressed by $v = (-1)A + 1B$. Our approach uses a sub-type of mass point for vector.

First, we define a function, namely *isVector*, of type $MassPoint \rightarrow Prop$ that takes a mass point value and returns true if v is a representation of a vector. In other words, there are two points A, B such that $v = (-1)A + 1B$. This function is then used as a filter in the following definition of vector.

```
Definition isVector (v:MassPoint):=
exists A, B :Point, v = (−1)A + 1B.

Record Vector : Type :=
          vecCons { mpOf :MassPoint; proof:isVector mpOf}.
```

Vector is defined by the record containing two fields. The first one *mpOf* is a value of mass point. The second is a proof showing that the first is a representation of a vector. We use the pair (mpOf, proof) to denote vector. Note that we use here the notion of dependent type in Coq where the type of some argument to a function depends on the value of other arguments. Indeed, the element *proof* has the type of *isVector mpOf* (essentially $\exists AB$ *:Point, mpOf = (-1)A + 1B*), hence its type depends on the value of *mpOf*.

Constructor for vector

The first functions that we have to consider after having created the type *Vector* are constructors. For a vector, we only have one constructor in the form of *vector A B*. Defining this constructor is simple by the mass point value $(-1)A + 1B$ and the trivial proof p showing this value is a representation of a vector. In particular, there exist two points A' and B' such that $(-1)A + 1B = (-1)A' + 1B'$.

Definition of operators

We now define algebraic operators of vectors. It is easy to find that we can define these operators by using their correspondence in mass point. In particular, we define the addition operator of two vectors by constructing the vector addition from the sum of mass point values of these vectors.

```
Definition add_Vect (v1 v2 : Vector ) :=
  match v1, v2 with  (mpOf1, proof_mpOf1) , (mpOf2, proof_mpOf2)
  => ((mpOf1 + mpOf2),(sumVecs_isVec mpOf1 mpOf2))
  end.
```

This definition is interpreted as follows: v1 and v2 have the type *Vector*, suppose that they are represented by records (mpOf1,proof_mpOf1) and (mpOf2, proof_mpOf2). We then construct a record of type *Vector* that takes $(mpOf1 + mpOf2)$ as the mass point value and consider it as the vector addition. The second element of this vector

is a proof showing that $(mpOf1 + mpOf2)$ is a representation of vector. This proof is provided by the function *sumVecs_isVec* constructed using *isVector* as follows

```
Lemma sumVecs_isVec :
 forall (value1 value2 :MassPoint),
  isVector value1 -> isVector value2 ->
  isVector ( value1 + value2).
```

Once this lemma is proved, its application with arguments mpOf1 and mpOf2 (*sumVecs_isVec mpOf1 mpOf2*) gives us an object with the type *isVector (mpOf1 + mpOf2).* Thus we can use this as the second element of the vector resulting from the addition.

.....Begin of Technical Details..... 1.
To prove this lemma, we first introduce hypotheses and destruct the hypotheses *isVector value1* and *isVector value2* to have 4 points A, B, C and D such that

$$value1 = (-1)A + 1B$$

and

$$value2 = (-1)C + 1D.$$

```
Proof.
intros value1 value2 H0 H1.
destruct H0 as [A [B H0]].
destruct H0 as [C [D H1]].

H0: value1 = (-1)A+1B
H1: value2 = (-1)C+1D
------------------------------------(1/1)
isVector ( value1 + value2)
```

By unfolding *isVector* in the goal and rewriting value1 and value2 by their representation of points, we have to prove $\exists A'\ B', (-1)A' + 1B' = (-1)A + 1B + (-1)C + 1D$. Observe that, if we choose $A' = A$, $(1B')$ can be calculated by $(-1)A + 1B' = (-1)A + 1B + (-1)C + 1D$, hence $1B' = 1B + (-1)C + 1D$.

We recall that with 2 mass points aA and bB where $a+b \neq 0$, we can construct their barycenter such that $\forall (ab : R)(AB : Point), a + b \neq 0 \rightarrow (a + b)(barycenter\ aA\ bB) = aA + bB$.

The sum of masses in the right hand side of the above equality differs from 0. Therefore, it is evident that we can construct B' by using the barycenter. The construction of B' is as follows : $1B' = 1B + (-1)C + 1D \Rightarrow 1B' = (-1)C + (1B + 1D)$ $\Rightarrow 1B' = (-1)C + (2(barycenter 1B 1D))$(apply the property of barycenter with 1B

and 1D) $\Rightarrow 1B' = 1(barycenter(-1)C(2(barycenter1B1D)))$ (apply the property of barycenter with (-1)C and 2(barycenter 1B 1D)).

The proof of existence of A'and B'is performed by respectively assigning A and (barycenter (-1)C (2(barycenter 1B 1D))) to them. These assignment are performed in Coq by tactic *exists* as follows

```
Proof.
..
rewrite H0,H1.
unfold isVector.
exists A.
exists (barycenter ((−1)C) (2(barycenter (1B) (1D)))).

H0: value1 = (−1)A+1B
H1: value2 = (−1)C+1D
______________________________________(1/1)
(−1)A+1(barycenter (−1)C (2(barycenter 1B 1D))=(−1)A+1B+(−1)C+1D
```

To prove this new goal, we use the property of barycenter in inverse direction with the above construction of B'. We make occurrences of barycenter disappear by repeatedly replacing $(a+b)(barycenter\ aA\ bB)$ with $aA+bB$. The proof obligation becomes $(-1)A+((-1)C+(1B+1D))=(-1)A+1B+(-1)C+1D$. This is solved by the tactic RingMP mentioned above.

```
Proof.
..
repeat rewrite barycenter_prop;auto.
Ring_MP.
Qed.
```

.....End of Technical Details.....

Similarly, we define the multiplication of vector.

```
Lemma multVec_isVec :
  forall (k:R) (mpOf :PP),
   isVector mpOf ->
   isVector (mult_PP k mpOf).
Definition mult_Vect (k :R) (v1 : Vector ) :=
  match v1 with
     vecCons mpOf proof_mpOf => vecCons (multVec_isVec k mpOf)
  end.
```

Tactic to translate calculations of vectors

By these operator definitions, calculations of vectors are essentially performed with their mass point value. However, to be able to reuse automatic calculations of mass

points, we need to translate equalities of vector in the proof context to equalities of mass points. The following tactic is designed to do this in Coq:

```
Ltac transf_Vector_MassPoint :=
(*step 1 : convert to structure*)
 repeat
  match goal with
  |  |-context [vector _ _] => unfold vector
  |  |-context [add_Vect _ _] => unfold add_Vect
  |  |-context [mult_Vect _ _]=> unfold mult_Vect
  | H: context [vector _ _] |- _ => unfold vector in H
  | H:context [add_Vect _ _] |- _=> unfold add_Vect in H
  | H:context [mult_Vect _ _]|- _=> unfold mult_Vect  in H
  end;
(*step 2 : translate structure to mass point*)
repeat
  match goal with
  | |- vecCons ?a = vecCons ?b =>
    rewrite <- MassPoint_Vector_equiv;simpl
  | H: vecCons ?a = vecCons ?b |- _=>
    rewrite <-MassPoint_Vector_equiv in H;simpl in H
  | _ => idtac
  end
).
```

The tactic is performed in 2 steps. In the first step (corresponding to the first repeat loop), we convert all terms of type *Vector* into their structure. We only have three functions that return an element of type *Vector* and these are converted as follows

- The constructor of vector *vector A B* is converted to ((-1)A+1B, _)
- The addition function *add_Vect u v* is converted to (mpOf u + mpOf v, _)
- The multiplication function *mult_Vect k v* is converted to (k (mpOf v), _)

These conversions are performed repeatedly. As a result, an equality of two terms of type *Vector* is replaced by an equality of calculated structures that represent these terms. For example, $\overrightarrow{w} = k\overrightarrow{u} + l\overrightarrow{v}$ is converted to $(mpOf\ w, \_) = ((k(mpOf\ u) + l(mpOf\ v)), \_)$.

In the second step (corresponding to the second repeat loop), the equalities of structure are translated to ones of their corresponding mass point by using the following equivalence

```
MassPoint_Vector_equiv : forall v1 v2 : Vector,
mpOf v1 = mpOf v2 <-> v1 = v2
```

In Coq the equality of two objects of type *Record* is only ensured by equality of all their components $mpOf\ v1 = mpOf\ v2 \wedge proof\ v1 = proof\ v2 \leftrightarrow (mpOf\ v1, proof\ v1) = (mpOf\ v2, proof\ v2) \leftrightarrow v1 = v2$. So, to prove this lemma, we have to use the irrelevant proof in Coq, which shows that all proofs of a property are equivalent.

```
Axiom proof_irrelevance : forall (P:Prop) (p1 p2:P), p1 = p2.
```

Using this equivalence, the tactic finds equalities of vector structure in the proof context, and repeatedly replaces them by equalities of mass points. For example, after doing this step, $\overrightarrow{w} = k\overrightarrow{u} + l\overrightarrow{v}$ is replaced by $mpOf\ w = (k(mpOf\ u) + l(mpOf\ v)$.

Using this tactic allows us to easily prove algebraic properties of vector. Indeed, we only need to use the tactic *transf_Vector_MassPoint* to get equalities of mass point and the automatic tactics on mass points to perform calculations. For example, the proof of distributivity for vectors is short as follows:

```
Lemma multVect_addVect_distributive: forall (r:R) (v1 v2 :Vector),
  mult_Vect r (add_Vect v1 v2) =
      add_Vect (mult_Vect r v1)(mult_Vect r v2).
Proof.
intros.
destruct v1;destruct v2.(*to represent vector by points*)
transf_Vector_MassPoint.
RingMP.
Qed.
```

The last properties of vector that we would like to present concern creating new point as barycenters. In the proofs of the lemma *sumVecs_isVec*, we can see somewhat the relation between them. Indeed, we saw the analysis to find out B' such that $(-1)A + 1B' = (-1)A + 1B + (-1)C + 1D$. This analysis leads us to construct B' using barycenter $B' = (barycenter(-1)C(2(barycenter\ 1B\ 1D)))$. In other words, we can construct B' such that $\overrightarrow{AB'} = \overrightarrow{AB} + \overrightarrow{CD}$.

In a generalized form of this, a question is that, with any point M and algebraic combination of vectors, if we can construct a point N such that $\overrightarrow{MN}$ equals this combination. The answer is yes by constructing N with the help of the barycenter function using a similar analysis. The following lemmas state the existence of such an N for some combinations. Note that we use operator addition and multiplication instead of *add_Vect* and *mult_Vect*

```
Lemma existence_multVectRepresentative :
 forall (M A B : Point) (k:R),
 {N | MN = k AB}.

Lemma existence_addVectRepresentative :
```

```
forall (M A B C D  :Point),
{N | MN⃗ = AB⃗ + CD⃗}.

Lemma existence_linearCombVectRepresentative:
forall (A B C D M :Point) (k1 k2 :R),
{N | MN⃗ = k1 AB⃗ + k2 CD⃗}.
```

Note that the statements of the lemmas use *constructive existence*. Roughly speaking, {x:A|P x} is a type and an instance of this type gives us an object satisfying the property P. This differs from $\exists x, Px$ that renders a proof. The first form is stronger than the second form. Indeed, if we can construct an object satisfying the property P, we can obviously prove the existence of an object satisfying the property P by using the newly constructed object.

As a result, proofs of these lemmas have to be performed in a constructive way. It means that we have to give a construction for N satisfying the requirements of the lemmas. For example, in the lemma *existence_multVectRepresentative*, we need to construct N such that $\overrightarrow{MN} = k\overrightarrow{AB}$. We have $(-1)M + 1N = k((-1)A + 1B) \rightarrow 1N = (-k)A + kB + 1M \rightarrow 1N = 1(barycenter\ (-k)A(k+1)(barycenter\ kB\ 1M))$, hence $N = (barycenter\ (-k)A(k+1)(barycenter\ kB\ 1M))$. This assignment of N allows us to readily prove that $\overrightarrow{MN} = k\overrightarrow{AB}$ by calculations of mass point.

We will later see the role of these properties in constructing Cartesian coordinate systems, defining midpoint, and defining some plane transformations.

2.4.2 Formalization of line

The notion of vector allows us to formalize the other notions in affine geometry (see Fig. 2.2). The following is a short presentation about it.

- The middle of two points and the gravity center of a triangle are defined using barycenters. In particular, the midpoint I of A and B is defined as the barycenter of the mass points 1A and 1B,

  $$midpoint\ A\ B := barycenter\ 1A\ 1B.$$

  The gravity center G of triangle ABC is defined by barycenter of 1A 1B and 1C,

  $$gravity\ A\ B\ C := barycenter\ 1A\ 2(barycenter\ 1B\ 1C).$$

  We can prove its properties such as: $G \in CI$ and $\overrightarrow{AG} = \frac{2}{3}\overrightarrow{AI}$ when I is the midpoint of BC, 3 medians of triangle intersect at G, etc.

- The parallelism of vectors is defined by collinearity. The definition is as follows $v1 \parallel v2 := v2 \neq \overrightarrow{0} \wedge \exists k : R, k \neq 0 \wedge \overrightarrow{v1} = k\overrightarrow{v2}$.

- The collinearity of three points is defined by using collinearity of vectors as follows

$$col\ A\ B\ C := A = B \vee \exists k : R, \overrightarrow{AC} = k\overrightarrow{AB}.$$

- Coplanarity is defined by showing that a vector is a combination of other vectors. In particular, coplanarity of A, B, C, and D is defined by

$$coplanar\ A\ B\ C\ D := \exists(k1\ k2 : R), \overrightarrow{AD} = k1\overrightarrow{AB} + k2\overrightarrow{AC}.$$

- Intersection of lines is defined by the existence of a common point. Relations between intersection, coplanarity and parallelism are introduced, for example

$$intersect\ (line\ AB)(line\ CD) \rightarrow coplanar\ A\ B\ C\ D,$$

$$\neg parallel\ (line\ AB)(line\ CD) \rightarrow intersect\ (line\ AB)(line\ CD) \vee \neg coplanar\ A\ B\ C\ D.$$

Notions in affine geometry are formalized and their properties are also proved. We now pay attention to the notions of parallelism of lines that are the heart of affine geometry. We present our formalization by comparing it with the one realized in the library of F. Guilhot, this allows readers to see the advantages of our approach and how we eliminate axioms. F. Guilhot considered lines as an abstract object, there is a function to construct a line from two points and axioms allowing to manipulate this type of line.

```
Variable line : Point -> Point -> Line.
Axiom line_permute : forall A B : Point,
 A <> B -> line A B = line B A.
Axiom colinear_line : forall A B C : Point,
 A <> B -> A <> C -> colinear A B C -> line A B = line A C.
```

In a Coq view, these axioms are strong because they state equalities of objects. Evidently, properties are preserved from one to the other. Therefore, using these axiom make the system weaker. Moreover, this formalization only provides only one construction of line using different points. Two other types of line, line passing through a point and perpendicular with a given line (denoted tline) and line passes through a point and parallel with a given line (denoted pline), were not approached in the library of F. Guilhot. Adding these would lead to auxiliary axioms. Besides, the lack of common structure that represents line raises difficulties in defining relations of lines, for example, parallelism of two lines have to be precisely defined for each type of line.

To fill these gaps, we need to devise a common data structure for all types of line. We then define operators and relations concerning lines for this structure. Line constructors corresponding the types of line are also introduced. Finally, properties of line is verified with this formalization.

Data structure for line

We chose to represent lines with a root point and a non zero direction vector. In Coq, this is expressed by a record with three elements, where the first is a point, the second is a vector, and the last is the proof that the vector is non zero. Once again, the structure *Record* with a dependent type is used.

```
Record Line:Type := lineCons
{rootOf:Point; vecOf : Vector; proofNonZero: isNonZeroVec vecOf }.
```

Definition of operators and relations

Equality: As done with vector, the first notion we need to define is the equality of lines. We observe that, in the case that two lines have the same root point, they are equal if the two direction vectors are collinear. In the other case, two lines are equal if the vector composed by the two root points and the two direction vectors are collinear. The equality of two lines $a = (A, \overrightarrow{u})$ and $b = (B, \overrightarrow{v})$ is defined by

$$a == b := \begin{cases} \overrightarrow{u} \parallel \overrightarrow{v} & \text{if } A = B \\ \overrightarrow{u} \parallel \overrightarrow{v} \parallel \overrightarrow{AB} & \text{if } A \neq B \end{cases}$$

In Coq this is expressed by

```
Definition lineEqual: (a b :Line):=
 match a, b with (A, u), (B, v)
=> (A = B → u ∥ v) ∧ (A ≠ B → (AB ∥ u ∧ u ∥ v).
```

We here remark that notions of "a==b" and "a=b" are different in Coq. The first expresses equivalence of lines, it is mapped to semantic equality of lines in geometry. It is coarser than the second notion which expresses equality of line at the level of data structures. This distinction will be clarified in the discussion section at the end of this chapter.

Parallelism: The representation of lines using a direction vector makes it easier to define the parallelism of lines. In fact, the parallelism of lines is defined by the parallelism of their direction vector. The parallelism of vectors is already defined. With $a = (A, \overrightarrow{u})$ and $b = (B, \overrightarrow{v})$, we define $a \parallel b := \overrightarrow{u} \parallel \overrightarrow{v}$.

A point lying on a line: a point lies on a line if and only if the vector, composed by this point with the root point of the line, and the direction vector are collinear. With $a = (A, \overrightarrow{u})$ and a point M, we define $M \in a := \overrightarrow{AM} \parallel \overrightarrow{u}$.

Definition of constructors

In this session, we present only 2 types of line: line with 2 points and parallel line. The remaining kind (perpendicular) is only formalized in Euclidean geometry where we have the notion of perpendicularity.

A line passing through two different points A and B is constructed by the root point A and the direction vector $\overrightarrow{AB}$. In the degenerate case where A and B coincide,

we return an arbitrary value for this line. The principle of excluded middle is used to determine cases of A and B. This is represented through a function *AB_decidable* that gives us A=B or A<>B.

```
Lemma isNonZeroVecCondition : forall A B :Point,
 A<>B ->isNonZeroVec (vector A B).
Parameter arbitraryLine :Line.
Definition line (A B :Point):=
 match (@AB_decidable (A = B)) with
  | left H => arbitraryLine
  | right H => lineCons A (vector A B)
               (isNonZeroVecCondition(A:=A)(B:=B) H)
end.
```

The chosen structure representing line emphasizes the role of vectors. The role of the second point in formalization of F. Guilhot is blurred. Using the direction vector makes it easier to formalize the line passing through a point and perpendicular with a given line (denoted tline) and the line passes through a point and parallel with a given line (denoted pline). Indeed, if we want to construct a line passing a point A and parallel with a given line, we can always use A as the root point of this line. As a result, defining a line leads to constructing its direction vector.

For the pline l_p passing a point A and is parallel with a given line $a = (B, \overrightarrow{u}, prf)$, it is easy to find that $\overrightarrow{u}$ can be used as the direction vector of l_p. The element *prf* that is a proof of $\overrightarrow{u} \neq \overrightarrow{0}$ is also reused. Thus, the parallel line is defined by $l_p := (A, \overrightarrow{u}, prf)$.

```
Definition lineP (A:Point) (a:Line) :=
 match a with lineCons B u prf
 => lineCons A u prf
 end.
```

Verification of properties

Our formalization allows us to verify properties of lines. The first ones that we want to cite here are the properties stated as axioms in the library of F. Guilhot.

```
Lemma align_permute: forall A B  : Point,
 A <> B -> line A B == line B A.
Lemma align_line: forall A B C : Point,
 A <> B -> A <> C ->
 col A B C -> line A B == line A C.
```

The first lemma can be proved easily. For the second one, we start with the hypotheses $A \neq B$ and $A \neq C$ we have that $\overrightarrow{AB} \neq \overrightarrow{0}$ and $\overrightarrow{AC} \neq \overrightarrow{0}$. By the definition of line $\overrightarrow{AB}$ and $\overrightarrow{AC}$ are direction vectors of the line AB and the line AC respectively. By the definition of collinearity (col A B C) we have that $\overrightarrow{AB}$ and $\overrightarrow{AC}$ are collinear. Thus $line\ AB == line\ AC$ by the definition of equality of lines for the case where lines have the same root point.

Other properties of lines are also proved. They include: the transitivity of parallelism $\forall a\ b\ c, \parallel\ b \wedge b\ \parallel\ c \rightarrow a\ \parallel\ c$; properties of parallel lines such as $\forall(A : point)(l : Line), pline(A\ l) \leftrightarrow A \in l \wedge pline(A\ l)\ \parallel\ l$; properties related to lying on a line such as $\forall A\ B, A \neq B \wedge A \in a \wedge B \in a \rightarrow line\ A\ B == a$ and $\forall A\ B\ C, A \neq B \wedge C \in line\ A\ B \rightarrow col\ A\ B\ C$.

2.5 Formalization of Euclidean geometry

In affine geometry, there is no way to talk about distances, orthogonality and angles. Euclidean geometry is built from affine geometry by adding axioms about the notion of scalar product (denoted by $\overrightarrow{u} \cdot \overrightarrow{v}$). This is also called dot product or inner product. The axioms of scalar products are introduced in Coq as follows

```
Variable scalarProduct : Vector -> Vector -> R.
Axiom scalarProduct_positive_Vector: forall v :Vector,
 scalarProduct v v >= 0.
Axiom scalarProduct_non_degenerate_Vector: forall v :Vector,
 scalarProduct v v = 0 -> v = ZeroVect.
Axiom scalarProduct_sym : forall u v :Vector,
 scalarProduct u v = scalarProduct v u.
Axiom scalarProduct_addVect_l: forall v1 v2 v3 :Vector,
 scalarProduct (add_Vect v1 v2) v3 =
 scalarProduct v1 v3 + scalarProduct v2 v3.
Axiom scalarProduct_multVect_l: forall (k : R)(u v :Vector),
 scalarProduct (mult_Vect k u) v = k * scalarProduct u v.
```

Where R is the type for real numbers. The scalar product in turn is used to define the followings entities

- **Length**: the length of a vector $\overrightarrow{v}$ is defined to be $|\overrightarrow{v}| = \sqrt{\overrightarrow{v} \cdot \overrightarrow{v}}$. In Coq, we have

  ```
  Definition magnitude ( v :Vector) :=
   sqrt (scalarProduct (v) (v)).
  ```

- **Normalization**: Given a nonzero vector $\overrightarrow{v}$. Normalization of $\overrightarrow{v}$ is a vector of unit length that points in the same direction as $\overrightarrow{v}$. We have $\overrightarrow{v} = \frac{\overrightarrow{v}}{|\overrightarrow{v}|}$. In Coq we have to use a function, namely *scalarVV_decidable vec1*, to decide that $\overrightarrow{v} \cdot \overrightarrow{v} = 0$ or $\overrightarrow{v} \cdot \overrightarrow{v} \neq 0$, these correspond to the case whether $\overrightarrow{v} = \overrightarrow{0}$ or not.

  ```
  Definition unitVector_representation (vec1:Vector) :=
  match (scalarVV_decidable vec1) with
   | left H => arbitraryUnitVect
   | right H => mult_Vect (/(sqrt ( scalarProduct vec1 vec1))) vec1
  end.
  ```

- **Distance between points**: the distance between two points is the length of the vector composed by these points, denoted by $|AB|$

```
Definition distance (A B: Point) :=
 sqrt (scalarProduct (vector A B) (vector A B)).
```

- **Orthogonality**: Given 2 vectors $\overrightarrow{u}$ and $\overrightarrow{v}$, we say that they are orthogonal if and only if their scalar product is equal to 0. In Coq, we have

```
Definition orthogonal (vec1 vec2 :Vector):=
 scalarProduct vec1 vec2 = 0.
```

Figure 2.3: Formalization of Plane Geometry

Other notions of Euclidean geometry are introduced one by one into the library. Our formalization is illustrated in Fig.2.3. We now focus on the formalization of some

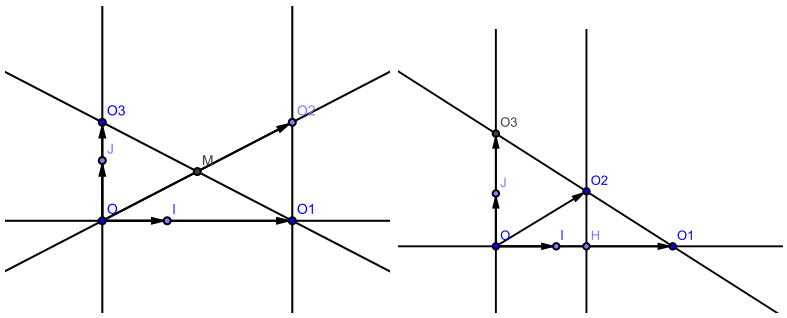
notion in plane geometry where the following axioms are used. These axioms assert the existence of 3 non-collinear points and their co-planarity with any fourth point.

- Axiom about existence of 3 not aligned points: there are 3 different and non-aligned points O, O_1 and O_2.
- Axiom about coplanarity: for any 4 points A, B, C and D in the plane, we always have that A, B and C are collinear or $\overrightarrow{AD}$ is linear combination of $\overrightarrow{AB}$ and $\overrightarrow{AC}$.

In fact, formalization of plane geometry is compatible with any plane defined by 3 non-collinear points. However, to make less complex statements and proofs in our library, we suppose the existence of 3 non-collinear points O, O_1 and O_2, and consider properties in this plane.

2.5.1 Cartesian coordinate system

The first notion whose formalization we want to present is a Cartesian coordinates system. This notion not only plays a crucial role in formalizations of many notions such as perpendicular line, trigonometry functions, signed area, etc, but also is a fundamental notion of algebraic geometry. A Cartesian coordinates system is represented by three non-collinear points that form two orthonormal vectors. In other words, they are two orthogonal vectors and both of them have unit length. The notions of orthogonality, distance, unit length that have been presented allow us to formalize this coordinate system. We present how to construct I, J, such that O, I and J form a Cartesian coordinate system, from three arbitrary non-collinear points O, O_1 and O_2. We now



(a) $OO_1 \perp O_1O_2$ (b) $OO_1 \not\perp O_1O_2$

Figure 2.4: Constructing a Cartesian coordinate system

have to construct I and J satisfying $\overrightarrow{OI} \cdot \overrightarrow{OJ} = 0$ (or $\overrightarrow{OI} \perp \overrightarrow{OJ}$), $|OI| = 1$ and $|OJ| = 1$. Let's consider the two configurations of O, O_1, O_2 in Fig. 2.4.
For the first case $\overrightarrow{OO_1} \perp \overrightarrow{O_1O_2}$, we construct O_3 such that $\overrightarrow{OO_3} = \overrightarrow{O_1O_2}$. From the hypothesis $\overrightarrow{OO_1} \perp \overrightarrow{O_1O_2}$ we have $\overrightarrow{OO_3} \perp \overrightarrow{OO_1}$.

For the second case $\overrightarrow{OO_1} \not\perp \overrightarrow{O_1O_2}$, Suppose that H is the orthogonal projection of O_2 on OO_1. O_3 is constructed from O_1, O_2 by

$$\overrightarrow{O_2O_3} = \frac{\overrightarrow{HO}}{\overrightarrow{HO_1}}\overrightarrow{O_2O_1} = \frac{\overrightarrow{O_2O}\cdot\overrightarrow{OO_1}}{\overrightarrow{O_2O_1}\cdot\overrightarrow{OO_1}}\overrightarrow{O_2O_1}$$

It's easy to find that this construction give us $\frac{\overrightarrow{O_1H}}{\overrightarrow{O_1O}} = \frac{\overrightarrow{O_1O_2}}{\overrightarrow{O_1O_3}}$. This leads to $\overrightarrow{OO_3} \parallel \overrightarrow{HO_2}$. As $\overrightarrow{O_2H} \perp \overrightarrow{OO_1}$, we get $\overrightarrow{OO_3} \perp \overrightarrow{OO_1}$.

For the both case, $O \neq O_1$ and $O \neq O_3$ are easily proved. This allows us to construct I and J by $\overrightarrow{OI} = \frac{1}{|OO_1|} \times \overrightarrow{OO_1}$ and $\overrightarrow{OJ} = \frac{1}{|OO_3|} \times \overrightarrow{OO_3}$ respectively. $\overrightarrow{OI}$ is a unit vector of $\overrightarrow{OO_1}$, therefore $|OI| = 1$. $\overrightarrow{OJ}$ is a unit vector of $\overrightarrow{OO_3}$, therefore $|OJ| = 1$. On the other hand, we have $\overrightarrow{OI} \perp \overrightarrow{OJ}$ by $\overrightarrow{OO_1} \perp \overrightarrow{OO_3}$. This Cartesian coordinate system is well formed.

Using this idea, the formalization in Coq is as follows:

```
Definition O3:= match twocaseofOO1O2 with
 |left _ =>
 proj1_sig (@existence_multVectRepresentative O O1 O2 1 )
 |right _ =>
 proj1_sig (@existence_multVectRepresentative O2 O2 O1 (O2O·OO1 / O2O1·OO1))
 end.
Definition I:=
   proj1_sig(@existence_multVectRepresentative O O O1 (1/|OO1|)).
Definition J:=
   proj1_sig (@existence_multVectRepresentative O O O3 (1/|OO3|)).
Definition orthonormal_cdnSys (O I J : Point) :=
OI·OJ = 0 ∧ OI·OI = 1 ∧ OJ·OJ = 1
```

Where, the function *twocaseofOO1O2* is to decider whether $\overrightarrow{OO1}\cdot\overrightarrow{O1O2} = 0$ or not. The application of *@existence_multVectRepresentative A B C k* is to construct a point D such that $\overrightarrow{AD} = k\overrightarrow{BC}$ as mentioned in page 21.

To verify if O, I and J form a Cartesian coordinate system, we have to prove that *orthonormal_cdnSys O I J*. We here focus only in the proof of $\overrightarrow{OI}\cdot\overrightarrow{OJ} = 0$ for the case $\overrightarrow{OO1}\cdot\overrightarrow{O1O2} \neq 0$. By construction of I and J, $\overrightarrow{OI}$ and $\overrightarrow{OJ}$ are normalizations of $\overrightarrow{OO_1}$ and $\overrightarrow{OO_3}$. This leads us to prove that $\overrightarrow{OO_1}\cdot\overrightarrow{OO_3} = 0$. The proof of this fact is different with our analysis in constructing O_3 where we use this reasoning $\frac{\overrightarrow{O_1H}}{\overrightarrow{O_1O}} = \frac{\overrightarrow{O_1O_2}}{\overrightarrow{O_1O_3}} \Rightarrow \overrightarrow{OO_3} \parallel \overrightarrow{HO_2} \Rightarrow \overrightarrow{OO_3} \perp \overrightarrow{OO_1}$. Our proof in Coq is realized by calculations of scalar products as follows.

.....Begin of Technical Details..... 2.

We recall some axioms about scalar product mentioned in page 26. We have:
$scalarProduct\_sym : \forall uv, \overrightarrow{u}\cdot\overrightarrow{v} = \overrightarrow{v}\cdot\overrightarrow{u}$,

$scalarProduct\_addVect\_l : \forall uvw, (\overrightarrow{u} + \overrightarrow{v}) \cdot \overrightarrow{w} = \overrightarrow{u} \cdot \overrightarrow{w} + \overrightarrow{v} \cdot \overrightarrow{w}$, and
$scalarProduct\_multVect\_l : \forall kuv(k\overrightarrow{u}) \cdot \overrightarrow{v} = k(\overrightarrow{u} \cdot \overrightarrow{v})$.

These state algebraic properties. As done with *Vector*, we can write a tactic to automatically simplify equations of scalar products and even decide their equalities.

```
Ltac RingScalarProduct :=repeat
rewrite scalarProduct_addVect_l ||rewrite scalarProduct_multVect_l
||rewrite scalarProduct_addVect_r ||rewrite scalarProduct_multVect_r;
try (ring ||ring_simplify).
```

The tactic transfer algebraic combinations in vector to ones in scalar product. As scalar product have a real value, equalities can be automatically simplified or decided using the tactic *ring_simplify* or *ring* respectively.

Let's return to our proof for the second case in Fig. 2.4(b). From the definition of O_3, we have $\overrightarrow{OO_3} = \overrightarrow{OO_2} + \overrightarrow{O_2O_3} = \overrightarrow{OO_2} + \frac{\overrightarrow{O_2O} \cdot \overrightarrow{OO_1}}{\overrightarrow{O_2O_1} \cdot \overrightarrow{OO_1}} \overrightarrow{O_2O_1}$. Using this equality, we have prove that

```
1 subgoal
OO1 · O1O2 ≠ 0
------------------------------------(1/1)
OO1 · (OO2 + (O2O · OO1)/(O2O1 · OO1) O2O1) = 0
```

Using the tactic *RingScalarProduct* to simplify this proof obligation, we have

```
RingScalarProduct.
```

```
1 subgoal
OO1 · O1O2 ≠ 0
------------------------------------(1/1)
OO1 · OO2 + (O2O · OO1)/(O2O1 · OO1) OO1 · O2O1 = 0
```

We normalize scalar products that appears in the goal by replacing $\overrightarrow{OO_1} \cdot \overrightarrow{OO_2}$ by $\overrightarrow{OO_2} \cdot \overrightarrow{OO_1}$ and $\overrightarrow{OO_1} \cdot \overrightarrow{O_2O_1}$ by $\overrightarrow{O_2O_1} \cdot \overrightarrow{OO_1}$ using the axiom *scalarProduct_sym*, and replacing $\overrightarrow{O_2O}$ with $(-1)\overrightarrow{OO_2}$. This gives us

```
RingScalarProduct.
repeat rewrite (@scalarProduct_sym OO1 OO2).
repeat rewrite (@scalarProduct_sym OO1 O2O1).
repeat replace O2O with ((-1)OO2) by RingVector.
```

```
1 subgoal
OO1 · O1O2 ≠ 0
------------------------------------(1/1)
OO2 · OO1 + ((-OO2) · OO1)/(O2O1 · OO1) O2O1 · OO1 = 0
```

We readily find that this equality can be solved by the tactic *field*. This tactic is similar to *ring*, but it is for fractions. Equality is decided in the condition that the denominator is not equal zero. Applying this tactic leads to the proof of $\overrightarrow{O_2O_1} \cdot \overrightarrow{OO_1} \neq 0$ that is easily solved.

```
RingScalarProduct.
repeat rewrite (@scalarProduct_sym OO1 OO2).
repeat rewrite (@scalarProduct_sym OO1 O2O1).
repeat replace O2O with ((-1)OO2) by RingVector.
field.
auto with geo.
Qed.

Proof completed.
```

.....End of Technical Details.....

Here, we use a database, namely *geo*, pour store useful proved theorems. This database is used to automatically prove simple goals.

The construction of this Cartesian coordinate system allows us to prove properties concerning representation of vectors by unit vectors. For example, we can prove that:

- Every vector can be represented by a combination of unit vectors:
  $\overrightarrow{MN} = (\overrightarrow{MN} \cdot \overrightarrow{OI})\overrightarrow{OI} + (\overrightarrow{MN} \cdot \overrightarrow{OJ})\overrightarrow{OJ}$.

- The representation is unique:
  $\overrightarrow{MN} = x\overrightarrow{OI} + y\overrightarrow{OJ} \rightarrow x = (\overrightarrow{MN} \cdot \overrightarrow{OI}) \wedge y = (\overrightarrow{MN} \cdot \overrightarrow{OJ})$.

- Calculations on vectors can be translated to calculations on coordinate values: suppose that $\overrightarrow{MN} = x\overrightarrow{OI} + y\overrightarrow{OJ}$ and $\overrightarrow{M'N'} = x'\overrightarrow{OI} + y'\overrightarrow{OJ}$, then we can conclude $\overrightarrow{MN} \cdot \overrightarrow{M'N'} = (x\overrightarrow{OI} + y\overrightarrow{OJ}) \cdots (x'\overrightarrow{OI} + y'\overrightarrow{OJ}) = xx' + yy'$.

Thus, apart from the use in constructing other notions, the formalization of Cartesian coordinate system and its properties makes it possible to introduce algebraic proofs into our system and paves the way to integrating algebraic systems into ours. This will be discussed in Chapter 4.

2.5.2 Elementary constructions

2.5.2.1 Perpendicular Line

Defining perpendicular line (denoted by l_t) that passes through a point A and perpendicular with a given line l is performed by a similar way as parallel line. We use A as the root point of perpendicular line. However, it is more complex in constructing the direction vector of l_t that is perpendicular with $\overrightarrow{v}$.

In fact, this challenge is overcome by constructing the notion of orthogonal vector in the Cartesian coordinate system (OIJ system). The orthogonal vector of $\overrightarrow{v}$ is a vector (denoted by $\overrightarrow{v}^{\perp}$) that is perpendicular and has the same magnitude as $\overrightarrow{v}$. Suppose $\overrightarrow{v}$ is represented in OIJ by $\overrightarrow{v} = x \times \overrightarrow{OI} + y \times \overrightarrow{OJ}$, we define $\overrightarrow{v}^{\perp}$ by $\overrightarrow{v}^{\perp} := -y \times \overrightarrow{OI} + x \times \overrightarrow{OJ}$. The definition in Coq is as follows

```
Definition orthoVect (v :Vector):=
add_Vect (mult_Vect (-scalarProduct v (vect O J)) (vect O I))
      (mult_Vect (scalarProduct v (vect O I))(vect O J)) .
```

Using calculations of scalar product, we can readily prove that $\overrightarrow{v} \cdot \overrightarrow{v}^{\perp} = 0$ and $\sqrt{\overrightarrow{v} \cdot \overrightarrow{v}} = \sqrt{\overrightarrow{v}^{\perp} \cdot \overrightarrow{v}^{\perp}}$, hence $\overrightarrow{v} \perp \overrightarrow{v}^{\perp}$ and $|\overrightarrow{v}| = |\overrightarrow{v}^{\perp}|$.

These properties allows us to use $\overrightarrow{v}^{\perp}$ as the direction vector of l_t. Therefore, we have this definition $l_t\ A\ l := (A, \overrightarrow{u}^{\perp})$ with $\overrightarrow{u}$ is the direction vector of l. In Coq, we have

```
Definition tline (A :Point)(l:Line):=
 lineCons A (vecOf l)⊥ (dirVec_isNonZeroVec (vecOf l)⊥).
```

2.5.2.2 Circle

Formalizing circles is simple. First, we use a type of records with two fields to define a circle: a center point and a radius.

```
Record Circle :Type
:= circleCons{ circleCenter :Point; radius :R}.
```

Other constructors of circles are defined by constructing a record of this type.

- Circle with a center point O and passing through a given point A is defined by

  ```
  Definition circle_Point (O:Point)(A:Point):=
  (circleCons O (distance O A)).
  ```

- Circle with diameter AB is defined by using the midpoint of AB as the center point and distance from this point to A as the radius.

  ```
  Definition circle_Diameter (A:Point)(B:Point):=
  (circleCons (midpoint A B) (distance (midpoint A B ) A)).
  ```

- Circle passing through 3 points A, B and C: to define this, we construct the perpendicular bisectors of AB and AC. We use the intersection point of these lines as the center point and distance from this point to A as the radius.

  ```
  Definition circle_3Points (A B C :Point):=
  let O:= intersectionPoint (perpBisector A B)(perpBisector A C) in
  (circle_Point O A)).
  ```

2.5.2.3 Intersection Point

Let's consider two lines l_1 and l_2. If $l_1 \parallel l_2$, they can be equal or have no common point. Therefore, in this section, we present how to constructiing the intersection point of 2 lines a and b only for the case where $\neg(l_1 \parallel l_2)$. Let's call M the intersection point of l_1 and l_2.

By definition of line, we can have the existence of four points A, B, C and D such that $A \neq B$, $C \neq D$, l_1 is represented by AB, and l_2 is represented by CD.

From the axiom about coplanarity of 4 points, we consider two cases as follows: The first case is where A, C and D are collinear. We easily see that the intersection point of a and b is A.

The second case is where A, C and D are not collinear and there exist two real numbers a and b such that

$$\overrightarrow{AB} = a\overrightarrow{AC} + b\overrightarrow{AD} \tag{2.1}$$

Since M is the intersection point of AB and CD, we have that M, C, and D are collinear, we can suppose that $\alpha\overrightarrow{MC} + \beta\overrightarrow{MD} = \overrightarrow{0}$. Replacing $\overrightarrow{MC}$ with $\overrightarrow{AC} - \overrightarrow{AM}$, $\overrightarrow{MD}$ with $\overrightarrow{AD} - \overrightarrow{AM}$, we have

$$(\alpha + \beta)\overrightarrow{AM} = \alpha\overrightarrow{AC} + \beta\overrightarrow{AD} \tag{2.2}$$

We also have the collinearity of M, A and B, hence $\exists k, \overrightarrow{AM} = k\overrightarrow{AB}$. Since A, C and D are not collinear, we can prove that $M \neq A$, hence $k \neq 0$. With equations 2.1 and 2.2, we can prove without much difficulty that $\alpha = k(\alpha+\beta)a$ and $\beta = k(\alpha+\beta)b$. This makes possible to perform deductions as $\alpha\overrightarrow{MC}+\beta\overrightarrow{MD} = \overrightarrow{0} \Rightarrow k(\alpha+\beta)a\overrightarrow{MC}+k(\alpha+\beta)b\overrightarrow{MD} = \overrightarrow{0} \Rightarrow a\overrightarrow{MC}+b\overrightarrow{MD} = \overrightarrow{0}$, hence $\overrightarrow{CM} = \frac{b}{a+b}\overrightarrow{CD}$. The last equation allows us to construct M. This is done by using the lemma *existence_multVectRepresentative* in page 21

2.5.3 Compound constructions

For compound constructions, we formalize them based on the way that they are built from primitive constructions. For example, instead of defining the orthogonal projection with three given points by axioms as mentioned in Section 2.2, we define an orthogonal projection of a given point C onto a given line a. We construct the line that passes through C and which is perpendicular with a, we prove that this line and a are not parallel, then we get the intersection point of them.

```
Definition orthogonalProjection(C :Point) (a :Line) :=
 intersectionPoint (lineT C a) a.
```

As we said above, students are familiar with definition of geometric objects by axioms about their properties. So, to avoid losing the pedagogical meaning of the library and to verify if objects are well formalized, we keep the former axioms in the form of theorems,

and prove them. Each object is thus accompanied by a pair of theorems. One theorem allows us to get properties from its definition, the other is to get a definition from properties.

```
Lemma orthogonalProjection_Properties :
 forall A B C H :Point,
 A <> B->
 H = orthogonalProjection C (line A B) ->
 collinear A B H /\ orthogonal (vect A B) (vect H C).
Lemma properties_orthogonalProjection :
 forall A B C H : Point,
 A <> B ->
 collinear A B H ->
 orthogonal (vect A B) (vect H C) ->
 H = orthogonalProjection C (line A B).
```

Proving theorems of the first kind is simpler than proving theorems of the second kind. The latter usually leads to a proof of uniqueness for the defined objects. Many compound constructions are introduced: the perpendicular bisector of two given points, the orthogonal projection of a point onto a line, the circumcircle of three given points, the center of a circumcircle, the orthocenter, the center of gravity,...

2.5.4 Angle and Trigonometry

Enriching scalar product makes it possible to measure angles. In fact, the cosine function of the oriented angle between two vectors $\overrightarrow{u}$ and $\overrightarrow{v}$ is defined by the following formula $\cos \widehat{\overrightarrow{v_1}\overrightarrow{v_2}} = \frac{\overrightarrow{v_1}\cdot\overrightarrow{v_2}}{|\overrightarrow{v_1}|\times|\overrightarrow{v_2}|}$

The sine function is defined with the support of the trigonometric identity $\sin \widehat{\overrightarrow{v_1}\overrightarrow{v_2}} = \cos(\frac{\pi}{2} - \widehat{\overrightarrow{v_1}\overrightarrow{v_2}})$. Let's consider our Cartesian coordinate system OIJ, we are allowed to chose $\widehat{\overrightarrow{OI}\overrightarrow{OJ}}$ to represent the angle $\frac{\pi}{2}$. With the notion of orthogonal vector mentioned above, we have $\widehat{\overrightarrow{v}\overrightarrow{v}^{\perp}} = \widehat{\overrightarrow{OI}\overrightarrow{OJ}} = \frac{\pi}{2}$ for every $\overrightarrow{v}$. As a result, we have $\sin \widehat{\overrightarrow{v_1}\overrightarrow{v_2}} = \cos(\frac{\pi}{2} - \widehat{\overrightarrow{v_1}\overrightarrow{v_2}}) = \cos(\widehat{\overrightarrow{v_1}\overrightarrow{v_1}^{\perp}} - \widehat{\overrightarrow{v_1}\overrightarrow{v_2}}) = \cos \widehat{\overrightarrow{v_2}\overrightarrow{v_1}^{\perp}} = \cos \widehat{\overrightarrow{v_1}^{\perp}\overrightarrow{v_2}}$.

Replacing the definition of the cosine function, we get that the definition of the sine function is given by scalar product as follows $\sin \widehat{\overrightarrow{v_1}\overrightarrow{v_2}} = \frac{\overrightarrow{v_1}^{\perp}\cdot\overrightarrow{v_2}}{|\overrightarrow{v_1}^{\perp}|\times|\overrightarrow{v_2}|}$

Defining trigonometric functions seems to be straightforward. However, since angles are input values for all trigonometric functions, we need to have a formalization of oriented angles before giving formal definition of the trigonometry functions and proving their properties.

Let's consider the set of axioms on the notion of oriented angle in the library of F. Guihot. As usual, oriented angles of vectors and operations on them are declared as primitive notions. Some axioms are then introduced to state their properties. The following axioms are added for this purpose.

- Axiom unitVector_eqAngle :With $A \neq B \wedge C \neq D$, if $\overrightarrow{A_1B_1}$ and $\overrightarrow{C_1D_1}$ are respectively unit vectors of $\overrightarrow{AB}$ and $\overrightarrow{CD}$, so $\widehat{\overrightarrow{AB}\overrightarrow{CD}} = \widehat{\overrightarrow{A_1B_1}\overrightarrow{C_1D_1}}$
- Axiom Chasles : $A \neq B \wedge C \neq D \wedge M \neq N \rightarrow \widehat{\overrightarrow{AB}\overrightarrow{CD}} + \widehat{\overrightarrow{CD}\overrightarrow{MN}} = \widehat{\overrightarrow{AB}\overrightarrow{MN}}$
- Axiom invAngle: $A \neq B \wedge C \neq D \rightarrow \widehat{\overrightarrow{AB}\overrightarrow{CD}} = -\widehat{\overrightarrow{CD}\overrightarrow{AB}}$
- Axiom exists_representation: for every vector angle α and a non-zero vector $\overrightarrow{u}$, there exists $\overrightarrow{v}$ such that $\widehat{\overrightarrow{u}\,\overrightarrow{v}} = \alpha$.

The fist axiom is to define oriented angles. The next two are for addition and inversion operation. To eliminate these axioms, as done for the formal description of lines, formalization of angles progresses in four steps. We need to provide a data structure to represent oriented angles. We give definition of oriented angles of two vectors. We then define operations. This formalization is verified by proving axioms and other properties of angles.

Data structure for angle

For the first axiom, by observing definitions of sin and cos functions, we find that $\sin \widehat{\overrightarrow{AB}\overrightarrow{CD}} = \sin \widehat{\overrightarrow{A_1B_1}\overrightarrow{C_1D_1}}$ and $\cos \widehat{\overrightarrow{AB}\overrightarrow{CD}} = \cos \widehat{\overrightarrow{A_1B_1}\overrightarrow{C_1D_1}}$. We also have a trigonometric property about the equivalence of angles $\widehat{\alpha} = \widehat{\beta} \leftrightarrow \sin\alpha = \sin\beta \wedge \cos\alpha = \cos\beta$. So defining oriented angle by a record of sin and cos is intuitively reasonable, and satisfies the first axiom.

In Coq, we use a record of 3 components (cs :Real, sn :Real, proof : $cs^2 + sn^2 = 1$). the first two elements are real numbers that express values of the sine and the cosine of the angle, while the last one is a proof ensuring that the first two are really trigonometric functions of this angle.

```
Record vectorAngle :Type :=
 angCons { cs :R; sn :R; _ : cs² + sn² = 1 }.
```

Definition of angle constructors

Using the above record to represent oriented angle, the oriented angle of 2 vectors $\overrightarrow{v_1}$ and $\overrightarrow{v_2}$ can be constructed by assigning values $\frac{\overrightarrow{v_1}\cdot\overrightarrow{v_2}}{|\overrightarrow{v_1}|\times|\overrightarrow{v_2}|}$ and $\frac{\overrightarrow{v_1}^{\perp}\cdot\overrightarrow{v_2}}{|\overrightarrow{v_1}^{\perp}|\times|\overrightarrow{v_2}|}$ to "cs" and "sn" elements respectively. The rest is a proof of $sn^2 + cs^2 = 1$. This equation is only true in the non-degenerate case where $|\overrightarrow{v_1}| \times |\overrightarrow{v_2}| \neq 0$ (i.e. $\overrightarrow{v_1} \neq \overrightarrow{0}$ and $\overrightarrow{v_2} \neq \overrightarrow{0}$). It is expressed by the following lemma in Coq.

```
Lemma sincosSquare :
 forall (v1 v2 :Vector),
|v1| × |v2| ≠ 0 -> (cs/(|v1|×|v2|))² + (sn/(|v1|×|v2|))² = 1
```

Because functions in Coq are total, every case needs to be considered. So, an arbitrary value is returned for the degenerated case where $|\overrightarrow{v_1}|\times|\overrightarrow{v_2}| = 0$ (i.e. $\overrightarrow{v_1} = \overrightarrow{0}$ or $\overrightarrow{v_2} = \overrightarrow{0}$).

$$\widehat{\overrightarrow{v_1}\,\overrightarrow{v_2}} = \begin{cases} arbitraryAngle & \text{if } |\overrightarrow{v_1}| \times |\overrightarrow{v_2}| = 0 \\ ((\frac{\overrightarrow{v_1}\cdot\overrightarrow{v_2}}{|\overrightarrow{v_1}|\times|\overrightarrow{v_2}|}), (\frac{\overrightarrow{v_1}^{\perp}\cdot\overrightarrow{v_2}}{|\overrightarrow{v_1}^{\perp}|\times|\overrightarrow{v_2}|}), sincosSquare) & \text{if } |\overrightarrow{v_1}| \times |\overrightarrow{v_2}| \neq 0 \end{cases}$$

The following is a definition in Coq. We use lemma "R_order_0" to compare $|\overrightarrow{v_1}|\times|\overrightarrow{v_2}|$ with 0, and use the structure "match..with" to treat derived cases.

```
Definition Angle (v1 v2 :Vector):=
 match (R_order_0 ( |v1| × |v2|)) with
|left H => abitraryAV
|right H => angCons (v1· v2)/(|v1|×|v2|) (v1⊥· v2)/(|v1⊥|×|v2|) (sincosSquare v1 v2 H)
end.
```

For every record expressing an angle, we always need to have the last component proving that the sum of square of the first two components equals 1. Conventionally, in this chapter, we sometimes omit this element and express angle by a record of only the first two. However, the proof is never ignored in Coq.

Definition of angle operators

To be able to manipulate the notion of oriented angles and prove trigonometric properties (such as the second and third axiom), we need to define operators and give definitions of special angles (such as zero angle, π, $\frac{\pi}{2}$, 2π, etc.). These can be derived from trigonometric properties.

Using symmetric properties $\sin(-\phi) = -\sin\phi$ and $\cos(-\phi) = \cos\phi$, we define the inversion of an angle $\phi = (cs, sn)$ by $-\phi = (cs, -sn)$.

Definition : $-_{angle}\phi := angCons(\cos\phi, -\sin\phi)$

Using angle sum identities $\cos(\alpha+\beta) = \cos\alpha\times\cos\beta - \sin\alpha\times\sin\beta$ and $\sin(\alpha+\beta) = \sin\alpha\times\cos\beta + \cos\alpha\times\sin\beta$, we define the sum of two angles (denoted by $+_{angle}$) as follows:

Definition : $\alpha +_{angle} \beta := angCons(\cos\alpha\times\cos\beta - \sin\alpha\times\sin\beta, \sin\alpha\times\cos\beta + \cos\alpha\times\sin\beta)$

Verification of angle properties

With these definitions, we can prove oriented angle properties, whereby axioms used to state these properties are eliminated (for example, the axioms in page 35). The proofs are usually derived from calculations and transformations of scalar product. We give here a such proof that is the one of Chasles relation for angle. This is stated as follows

$\forall\ \overrightarrow{u}\ \overrightarrow{v}\ \overrightarrow{w}, \overrightarrow{u} \neq \overrightarrow{0} \wedge \overrightarrow{v} \neq \overrightarrow{0} \wedge \overrightarrow{w} \neq \overrightarrow{0} \rightarrow \widehat{\overrightarrow{u}\,\overrightarrow{v}} +_{angle} \widehat{\overrightarrow{v}\,\overrightarrow{w}} = \widehat{\overrightarrow{u}\,\overrightarrow{w}}$

By the non-degenerate conditions, we can unfold the definition of angles and prove that:

$$(\frac{\overrightarrow{u}\cdot\overrightarrow{v}}{|\overrightarrow{u}|\times|\overrightarrow{v}|},\frac{\overrightarrow{u}^{\perp}\cdot\overrightarrow{v}}{|\overrightarrow{u}^{\perp}|\times|\overrightarrow{v}|})+_{angle}(\frac{\overrightarrow{v}\cdot\overrightarrow{w}}{|\overrightarrow{v}|\times|\overrightarrow{w}|},\frac{\overrightarrow{v}^{\perp}\cdot\overrightarrow{w}}{|\overrightarrow{v}^{\perp}|\times|\overrightarrow{w}|})=(\frac{\overrightarrow{u}\cdot\overrightarrow{w}}{|\overrightarrow{u}|\times|\overrightarrow{w}|},\frac{\overrightarrow{u}^{\perp}\cdot\overrightarrow{w}}{|\overrightarrow{u}^{\perp}|\times|\overrightarrow{w}|})$$

Using the definition of the addition operator, we can replace $(c1, s1) +_{angle} (c2, s2)$ with $(c1c2 - s1s2, c1s2 + c2s1)$. Applying this gives us
$(\frac{\overrightarrow{u}\cdot\overrightarrow{v}}{|\overrightarrow{u}|\times|\overrightarrow{v}|}\times\frac{\overrightarrow{v}\cdot\overrightarrow{w}}{|\overrightarrow{v}|\times|\overrightarrow{w}|}-\frac{\overrightarrow{u}^{\perp}\cdot\overrightarrow{v}}{|\overrightarrow{u}^{\perp}|\times|\overrightarrow{v}|}\times\frac{\overrightarrow{v}^{\perp}\cdot\overrightarrow{w}}{|\overrightarrow{v}^{\perp}|\times|\overrightarrow{w}|},\frac{\overrightarrow{u}\cdot\overrightarrow{v}}{|\overrightarrow{u}|\times|\overrightarrow{v}|}\times\frac{\overrightarrow{v}^{\perp}\cdot\overrightarrow{w}}{|\overrightarrow{v}^{\perp}|\times|\overrightarrow{w}|}+\frac{\overrightarrow{v}\cdot\overrightarrow{w}}{|\overrightarrow{v}|\times|\overrightarrow{w}|}\times\frac{\overrightarrow{u}^{\perp}\cdot\overrightarrow{v}}{|\overrightarrow{u}^{\perp}|\times|\overrightarrow{v}|})$
$=(\frac{\overrightarrow{u}\cdot\overrightarrow{w}}{|\overrightarrow{u}|\times|\overrightarrow{w}|},\frac{\overrightarrow{u}^{\perp}\cdot\overrightarrow{w}}{|\overrightarrow{u}^{\perp}|\times|\overrightarrow{w}|})$

Thus we have to prove 2 equalities

- $\frac{\overrightarrow{u}\cdot\overrightarrow{v}}{|\overrightarrow{u}|\times|\overrightarrow{v}|}\times\frac{\overrightarrow{v}\cdot\overrightarrow{w}}{|\overrightarrow{v}|\times|\overrightarrow{w}|}-\frac{\overrightarrow{u}^{\perp}\cdot\overrightarrow{v}}{|\overrightarrow{u}^{\perp}|\times|\overrightarrow{v}|}\times\frac{\overrightarrow{v}^{\perp}\cdot\overrightarrow{w}}{|\overrightarrow{v}^{\perp}|\times|\overrightarrow{w}|}=\frac{\overrightarrow{u}\cdot\overrightarrow{w}}{|\overrightarrow{u}|\times|\overrightarrow{w}|}$
- $\frac{\overrightarrow{u}\cdot\overrightarrow{v}}{|\overrightarrow{u}|\times|\overrightarrow{v}|}\times\frac{\overrightarrow{v}^{\perp}\cdot\overrightarrow{w}}{|\overrightarrow{v}^{\perp}|\times|\overrightarrow{w}|}+\frac{\overrightarrow{v}\cdot\overrightarrow{w}}{|\overrightarrow{v}|\times|\overrightarrow{w}|}\times\frac{\overrightarrow{u}^{\perp}\cdot\overrightarrow{v}}{|\overrightarrow{u}^{\perp}|\times|\overrightarrow{v}|}=\frac{\overrightarrow{u}^{\perp}\cdot\overrightarrow{w}}{|\overrightarrow{u}^{\perp}|\times|\overrightarrow{w}|}$

Proofs of these equalities are similar, we here consider the proof of the first one. In the formalization of orthogonal vector, we have $|\overrightarrow{i}| = |\overrightarrow{i}^{\perp}|$ for every $\overrightarrow{i}$. Using this to replace all occurrences of magnitudes of orthogonal vectors by ones of original vectors and using the tactic *field_simplify* in Coq allows us to simplify the equality. It remains to prove that

$$\frac{\overrightarrow{u}\cdot\overrightarrow{v}\times\overrightarrow{v}\cdot\overrightarrow{w}-\overrightarrow{u}^{\perp}\cdot\overrightarrow{v}\times\overrightarrow{v}^{\perp}\cdot\overrightarrow{w}}{|\overrightarrow{v}|\times|\overrightarrow{v}|}=\overrightarrow{u}\cdot\overrightarrow{w}$$

Let return to calculations of scalar product using coordinate values mentioned in Section 2.5.1. We use x_i and y_i to denote magnitude of $\overrightarrow{i}$ in the axes of the OIJ system, they are calculated by $x_i = \overrightarrow{i}\cdot\overrightarrow{OI}$ and $y_i = \overrightarrow{i}\cdot\overrightarrow{OJ}$. We have $\overrightarrow{i}\cdot\overrightarrow{j} = x_ix_j + y_iy_j$ and $\overrightarrow{i}^{\perp}\cdot\overrightarrow{j} = x_{i^{\perp}}x_j + y_{i^{\perp}}y_j = -y_ix_j + x_iy_j$ for every $\overrightarrow{i}$ and $\overrightarrow{j}$. These equalities are then repeatedly used to replace scalar products in our proof obligation by their corresponding value with coordinates. Moreover, we have $|\overrightarrow{v}|\times|\overrightarrow{v}| = \sqrt{\overrightarrow{v}\cdot\overrightarrow{v}}\sqrt{\overrightarrow{v}\cdot\overrightarrow{v}} = \overrightarrow{v}\cdot\overrightarrow{v}$. Therefore, the new proof obligation is as follows

$$\frac{(x_ux_v+y_uy_v)(x_vx_w+y_vy_w)-(-y_ux_v+x_uy_v)(-y_vx_w+x_vy_w)}{x_vx_v+y_vy_v}=x_ux_w+y_uy_w$$

This equality is solved using the tactic *field* in Coq. □.

Trigonometric functions and identities

For verification of trigonometric identities, the formalization of oriented angle allows us to define trigonometric functions by values of "cs" and "sn" elements. The formal definitions are follows.

```
Definition sin (angle:vectorAngle):= sn angle.
Definition cos (angle:vectorAngle):= cs angle.
```

Other trigonometric identities are verified without much effort. For example, by the definitions of the angle operators, the following trigonometric identities are easily verified

- $\sin(\alpha +_{angle} \beta) = \sin\alpha \times \cos\beta + \cos\alpha \times \sin\beta$
- $\sin(\alpha +_{angle} \beta) = \cos\alpha \times \cos\beta - \sin\alpha \times \sin\beta$
- $\sin(-_{angle}\phi) = -\sin\phi$
- $\cos(-_{angle}\phi) = \cos\phi$

2.5.5 Plane transformation

This part deals with the formalization of plane transformations including rotations, reflections, translations and dilatations. Existing axioms in this part of the library of F. Guilhot are used to define these notions. They are again in the form of defining notion from properties and getting properties from notion. The following example gives axioms for rotations. The first one is to get its properties and the next two are to define the notion.

```
Axiom  rotation_def : forall (I M N :Point)(a:AV),
I <> M -> N = rotation I a M ->
distance I N = distance I M /\ a = Angle (vector I M)(vector I N).
Axiom rotation_def1_centre :
 forall (I : PO) (a : AV), I = rotation I a I.
Axiom rotation_def2 : forall (I M N:PO) (a:AV),
I <> M -> distance I N = distance I M ->
a = Angle (vector I M) (vector I N) -> N = rotation I a M.
```

Eliminating axioms of this kind is reduced to constructing geometric objects image of the plane transformations. Among the transformations cited above, translation and dilatation are formalized without much effort. We only need to use an application of *existence_multVectRepresentative* (mentioned in page 21) that allows us to produce a point D from three points A, B and C and a real number k such that $\overrightarrow{AD} = k \times \overrightarrow{BC}$.

To construct the reflection of M across a line l, first, we construct the orthogonal projection H of this point on l. We then define the image point M by N that is constructed from $\overrightarrow{MN} = 2 \times \overrightarrow{MH}$.

Constructing the image point N of the point M with rotation of the center I and angle α is more complex. It relies on constructing a point P such that $\widehat{\overrightarrow{IM}\overrightarrow{IP}} = \alpha$. N. Then is constructed from I and P using $\overrightarrow{IN} = \frac{|\overrightarrow{IM}|}{|\overrightarrow{IP}|} \times \overrightarrow{IP}$.

```
Definition rotation (I:Point)(a:AV)(M:Point) :=
match (@eqPoint_decidable (I =M)) with
| left H => M
```

```
| right H =>
 let P:=proj1_sig(@existence_angle_represnt_general1 α I M) in
 proj1_sig (@existence_multVectRepresentative I I P (|IM|/|IP|)
end.
```

When definitions are given by constructions, axioms in the first form that gets properties from the notion are easily proved. Axioms in the second form that defines an object from given properties are more difficult, because we have to prove the uniqueness of the defined objects. Proof of the above axiom "rotation_def2" is an example. We need to prove that $|IN| = |IM| \wedge \alpha = \overrightarrow{IM}\overrightarrow{IN} \rightarrow N = rotation\ I\ \alpha\ M$. Let $M' = rotation\ I\ \alpha\ M$, from lemma *rotation_def* that is easily proved, we have $|IM'| = |IM| \wedge \alpha = \overrightarrow{IM}\overrightarrow{IM'}$. As a result, we have to prove that $N = M'$ while we have $|IN| = |IM'| = |IM| \wedge \overrightarrow{IM}\overrightarrow{IN} = \overrightarrow{IM}\overrightarrow{IM'} = \alpha$. This means that with the given properties, we only have one object satisfying them.

Using these transformations, composed transformations are then formalized. The notion of similar triangles and the properties about the ratio of their sides are also introduced.

2.6 Discussion

Constructiveness Our formalization is geometrically constructive. This relies not only on building compound geometric constructions from elementary ones in plane geometry, but also on constructive definitions for other geometric notions such as plane transformations. The existence of geometric objects is ensured by their construction.

Besides, our formalization respects the notion of constructive existence. This is expressed in stating existence of points related to given combinations of vector in page 21, where, the statements use constructive existence. For example, a lemma that is usually used in our proof is $\{\mathrm{N}|\overrightarrow{MN} = k\overrightarrow{AB}\}$. This force us into constructing N satisfying the condition in order to prove the lemmas. By this reason, vectors used in the combination of the statements are concretized by points. In particular, our example can not be stated as $\{\mathrm{N}|\overrightarrow{MN} = \overrightarrow{v}\}$. It is because we know that there exist two points A and B such that $\overrightarrow{v} = \overrightarrow{AB}$ but we can not get these points. We can do this after constructing the OIJ system, where, $\overrightarrow{v}$ is represented by $\overrightarrow{OA} = x_v\overrightarrow{OI} + y_v\overrightarrow{OJ}$. $x_v = \overrightarrow{v} \cdot \overrightarrow{OI}$ and $y_v = \overrightarrow{v} \cdot \overrightarrow{OJ}$ can be calculated, three points O, I and J exist, hence A is constructed. As a result, N is constructed from M, O and A.

However, the formalization is not totally constructive from the logical point of view. We still use a library in Coq about the law of excluded middle to assert equality or difference of two given points, relative position of points, etc. in proofs by cases or in some definitions of elementary objects. Furthermore, in the case of the intersection point of two lines, returning an arbitrary point as their intersection point when the two

lines are parallel makes our system more complex. We have to use a library that allows us to construct an object from the proof of its existence.

Equality and Equivalence The fact that we use user-defined structures (setoids) to represent elementary notions such as line, angle, etc. using vector allows us to eliminate a lot of axioms. Instead of adding axioms which assert equalities related to these notion, we prove corresponding lemmas about equivalences.

```
Axiom align_permute: forall A B : Point,
 A <> B -> line A B = line B A.
```

is eliminated and replaced by proving

```
Lemma align_permute: forall A B : Point,
 A <> B -> line A B == line B A.
```

However, this replacement makes the system lose some of its semantics. Equality ($x = y$) expresses an identity of Leibniz's law. It means that the objects have exactly the same properties $\forall P, (Px \leftrightarrow Py)$. This allows us to replace x by y in every geometric relations where there are occurrence of x.

On the other hand, equivalence ($x == y$) is a user-defined equality for setoids. the objects x and y only have semantic equivalence and are not the same. We can readily prove that equivalence relations of line, of angle have reflexivity, symmetry and transitivity that are also properties of equality. However, if we want that equivalent objects can be replaced as identical objects in geometric relations, we have to prove that these relations are preserved with this replacement.

Proving this is long because it is realized for every geometric relations having occurrence of line or angle such as : parallelism of line, perpendicularity of lines, intersection of lines, a point lying on a line, definition of intersection point, definition of orthogonal projection, sine and cosine functions, operators of angle, etc. For example, the following lemmas are for parallelism and incidence.

```
Lemma parallel_Line_compat :
 forall x x', x == x' ->
 forall y y', y == y' ->
 (parallel_Line x y    <->  parallel_Line x' y').

Lemma liesOnLine_compat :
 forall A :Point,
 forall x x' , x == x' ->
 (liesOnLine A x    <->  liesOnLine A x').
```

Finally, in order to use tactics such as *rewrite,replace* to replace an object by its equivalence in the geometric relations, we declare morphisms for these relations.

```
Add Morphism parallel_Line
 with signature  lineEqual ==>lineEqual ==>(@iff)
```

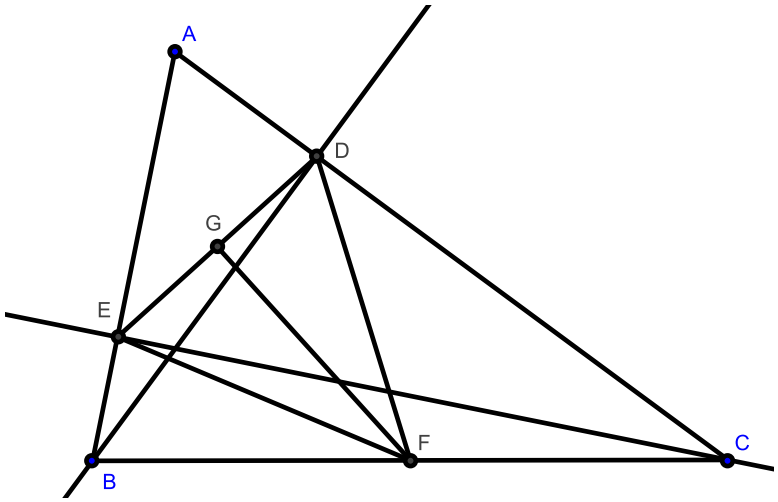


Figure 2.5: An example

```
 as parallel_Line_mor.
Proof.
 exact parallel_Line_compat.
Qed.
```

Comparison with the development of F. Guilhot Our development can be considered as providing an algebraic foundation and reformalizing the development of F. Guilhot. Needless axioms are eliminated, this makes the library cleaner from a logical point of view. As a result, the axioms system is reduced to 13 axioms that are really familiar with students

The fact that we provide vector based definitions for geometry notions makes our library more complicated. However, by proving properties of these notions, the pedagogical meaning of the library is preserved. Users can use geometry notions through their properties without knowing how they are defined. Furthermore, proofs of these properties use calculations of vector, scalar products, traditional methods, students are familiar with the notions hence proofs.

For other proofs in the development of F. Guilhot, there are changes arising from changes in definitions of geometry notions or from changes of statements. However, certain proof ideas are reused.

2.7 An example about school proof and formal proof

We now consider differences between proofs in school and formal proofs through the following example.

Example 1. : This example is illustrated in Fig. 2.5. Let BD and CE be two altitudes of triangle ABC and points G and F be the midpoints of BC and DE respectively. It

holds that GF $\perp$ DE.

2.7.1 The proof in school

To prove $GF \perp DE$, we observe that FG is the median of $\triangle FDE$ since G is midpoint of DE. If we have $\triangle FDE$ is isosceles at F, then FG is the altitude of $\triangle FDE$, hence $GF \perp DE$. As a result, we have to prove $GF \perp DE$, or that $|FD| = |FE|$. This deduction is insured by the following rule

Rule 1. : Given a $\triangle MNP$ and I is midpoint of NP. If $\triangle MNP$ is isosceles at M, then $MI \perp NP$.

Now, we have to prove the new goal $FD = FE$. We observe $\triangle EBC$, this is a right triangle with A the right angle. We remember that we have a rule about median of right triangle as follows

Rule 2. : Given a $\triangle MNP$ with M the right angle, and I is midpoint of NP. We have $|IM| = |IN| = |IP|$.

Because $\triangle DBC$ is a right triangle with the right angle in D and F is the midpoint of BC, we can apply this rule and we get $|FD| = |FB| = |FC|$.

By a similar way, we have $|FE| = |FB| = |FC|$.

From the last two equations $|FD| = |FB| = |FC|$ and $|FE| = |FB| = |FC|$, we can easily prove that $|FE| = |FD|$.

2.7.2 The formal proof

There are several ways to state this theorem in Coq (with predicate form and construction form). We use here the statement as follows

```
Lemma perpExample: forall A B C D E F G :Point,
¬col A B C ->
¬parallel (lineT B (line A C))(line A C) ->
D = intersectionPoint (lineT B (line A C))(line A C) ->
¬parallel (lineT C (line A B))(line A B) ->
E = intersectionPoint (lineT C (line A B))(line A B) ->
F = midpoint B C ->
G = midpoint E D ->
F <> G ->
D <> E ->
perpendicular (line F G)(line D E).
```

We note that some conditions to state that lines are not parallel and points are not equal are added to insure existence of the intersection points and lines in constructions. Deduction steps in proving this theorem are performed in Coq as follows

Step 1: We get properties from definitions of D and E in the hypotheses. The definition of D by the intersection point of line AC with *lineT* passing through B and

perpendicular with line AC gives us $D \in AC$ and $D \in (lineT\ B\ AC)$. The last property gives us $BD \perp AC$. This is insured by the following lemmas

```
Lemma intersectionPoint_lyingOn:
 forall (I : Point) (a b : Line),
 ¬ parallel_Line a b ->
 I = intersectionPoint a b ->
 liesOnLine I a /\ liesOnLine I b

Lemma lineT_property1 :
 forall (A B : Point) (a : Line),
 liesOnLine A (lineT B a) ->
 perpendicular (line A B) a
```

We use *intersectionPoint_lyingOnT* to get $D \in AC$ and $D \in (lineT\ B\ AC)$ from the definition of D. We then use *lineT_property1* to get $BD \perp AC$. By a similar way, we get properties from definition of E. These is performed in Coq as follows

```
intros.
destruct (@plane_intersectionPoint_lyingOn D (lineT B AC) AC);auto.
assert (H10:=@lineT_property1 D B (line A C) H8).

destruct (@plane_intersectionPoint_lyingOn E (lineT C AB) AB);auto.
assert (H13:=@lineT_property1 E C (line A B) H11).
```

We have the proof context:

```
H8 : liesOnLine D (lineT B (line A C))
H9 : liesOnLine D (line A C)
H10 : perpendicular (line D B) (line A C)
H11 : liesOnLine E (lineT C (line A B))
H12 : liesOnLine E (line A B)
H13 : perpendicular (line E C) (line A B)
______________________________________(1/1)
perpendicular (line F G) (line D E)
```

Step 2: We apply the the backward-chaining deduction using rule 1 with the configuration of F D E and G. There are 2 new goals

```
______________________________________(1/2)
¬ col F D E
______________________________________(2/2)
distance F D = distance F E
```

Step 3: The first goal $\neg\ col\ F\ D\ E$ is proved by contradiction. We suppose that F, D and E are collinear. With a long proof, we arrive the fact that F is midpoint of DE, hence $F = G$. This is contradiction with the hypothesis $F \neq G$. Thus, the goal is solved.

Step 4: Now we have to prove the second goal $|FD| = |FE|$. We do not directly prove this equality with backward-chaining method. We observe that rule 2 can be used for $\triangle DBC$ and F midpoint of BC. We consider applying the rule with the configuration of D, B, C and F, once its hypotheses being $DB \perp DC$ and F *is midpoint of* BC are verified, it gives us the new property $|FD| = |FB| = |FC|$.

Here, we mix backward and forward methods and write a tactic for this purpose.

```
Ltac lapply2 H:=
match type of H with
|?A->?B  =>
  let H' :=fresh in
  lapply H;[intros H';lapply2 H'| ];try clear H
|?A =>  idtac
end.
```

Suppose we have a base rule of the form
$\forall GeometricElements, Hyp_1 \wedge \cdots \wedge Hyp_n \rightarrow Goal$
The application of the tactic to this rule makes *Goal* become a new hypothesis of our context. But then we have to prove every hypothesis Hyp_1 of this rule.

Let's return to our proof. Rule 2 is used with this tactic as follows

```
lapply2 (@rule2_median_rightTriangle D B C F);auto with geo.
H15 : distance F D = distance F B /\ distance F D = distance F C

______________________________________(1/2)
distance F D = distance F E
______________________________________(2/2)
perpendicular (line D B) (line D C)
```

Step 5: This step is similar to Step 4, and is performed by applying rule 2 for the configuration of E, B, C and F. We have

```
lapply2 (@rule2_median_rightTriangle E B C F);auto with geo.

H15 : distance F D = distance F B /\ distance F D = distance F C
H16 : distance F E = distance F B /\ distance F E = distance F C

______________________________________(1/3)
distance F D = distance F E
______________________________________(2/3)
perpendicular (line E B) (line E C)
______________________________________(3/3)
perpendicular (line D B) (line D C)
```

Step 6: The first goal of Step 5 is easily proved. We decompose the hypotheses H15 and H16 to have $|FD| = |FB|$ and $|FE| = |FB|$. We then replace FD in the goal by FB, the goal becomes $|FB| = |FE|$. This is already obtained by composing H16.

```
decompose [and] H15 .
decompose [and] H16 .
rewrite H17;auto.
```

Step 7 We prove the second goal of Step 5 $EB \perp EC$. In fact, we have in hypotheses $H13 : EC \perp AB$. As a result, we can prove this goal by proving equality of lines $EB == AB$.

Step 8 This step is to proved $DB \perp DC$ is perform similarly to Step 7.

2.7.3 Discussion

By comparing both proofs, we see that the proof in Coq can follow up the school proof. However, there are many auxiliary goal that need to be proved. They come from non-degenerate conditions of applied rules. In our example, among 8 steps, there are only steps 1, 2, 4 and 5 that correspond to steps in the school proofs. Let's consider other steps and how to solve them.

In step 3, we have to prove that $\neg col\ F\ D\ E$. This is a special case of the figure where F coincides with G. In fact, this goal is proved with proof by contradiction. We suppose that F, D and E are collinear. We have prove a property $FD = FE$, by which we can prove that F is midpoint of DE. Hence we have contradiction.

It's easy to see that $FD = FE$ is the goal of the next step. It seems that we have to prove 2 times this goals with not very different context. In this case, it's better if we prove $FD = FE$ (or assert $FD = FE$ and prove after) before step 2 that generate the goal $\neg\ col\ F\ D\ E$.

In step 7 (as well in step 8), we have to prove that $EB \perp EC$. In fact, to work with line stated by 2 points, we always need to have the difference of these points. In particular, we need $E \neq B$ and $E \neq C$.

$E \neq C$ can be proved by the fact that A, B and C are not collinear. However, we can not prove that $E \neq B$. In fact, it is a degenerate case where $\triangle ABC$ is a right triangle at B. So, to solve this, somewhere before step 5 that require this difference, we have to use the proof by cases to consider 2 cases $B = E$ and $B \neq E$.

Once we have these difference, $EB \perp EC$ is proved by equivalence of EB and AB as mentioned in step 7. Indeed, we have $E \in AB$, $B \in AB$, so it is easy to have $EB == AB$. In the hypotheses we have $EC \perp AB$ (from definition of E), hence the goal is proved. Goals of this kind need to be proved automatically.

Therefore, the final proof of this theorem progresses in the following steps. the proof detail in Coq is presented in appendix A.

Part 1: Get properties from the hypotheses

Part 2: Get properties about difference, for example $B \neq D$, $C \neq E$

Part 3: We have $B = E$ and $C = D$. Replacing E by B and D by C in the hypotheses, we have $AB \perp BC$ and $AC \perp BC$. These allows us to deduce $AB \parallel AC$.

This is contradictor with the hypothesis $\neg col\ A\ B\ C$.

Part 4: We have $B = E$ and $C \neq D$. We prove this case as follows

- By $C \neq D$ and $B \neq D$, we use rule 2 for $\triangle DBC$ and F to have $|FB| = |FD|$.
- We apply rule 1 with F, D, E and G to have $FG \perp DE$
- We prove the condition of rule 1 $\neg col\ F\ F\ E$ by using $|FB| = |FD|$ of the first step

Part 5: We have $B \neq E$ and $C = D$. We prove this by a similar way to part 4.

Part 6: We have $B \neq E$ and $C \neq D$. The proof is as one mentioned above.

By comparing between the school proof and the formal proof, we find that, for formal proofs, we spend many efforts for degenerate cases, we need to have to a good proof strategies to avoid the same proof many times and we need to have automatic tactics for some easy proofs.

2.8 Conclusion and perspectives

With more than 20000 lines of formalization in about 50 files, we re-formalize the library of F. Guilhot by focusing into affine geometry and Euclidean geometry for plane. Our new system is based only on 13 axioms. Much of the redundancy of the axiomatic system is eliminated. Geometric notions are formalized in the manner that is closer to the student's knowledge and more constructive. Some new notions are also produced to enrich the library such as similar triangle, orientation, etc. The library allows to interactively produce traditional proofs.

Providing definitions of geometric notions, by showing how they are constructed from elementary geometric constructions, gives students a constructive view of geometry theorems in logic. This helps students to understand geometric constructions better.

Using proof assistant offers high level of confidence, but at the same time, it forces users to delve into detail which leads to very technical proofs. This is not adapted to their level of abstraction. Some automatic tactics were developed in this direction, but they are not enough to make proofs of the library natural. This is the first point that needs to be improved

Constructed objects are defined by functions which are total function in Coq. So, their properties can be obtained under some conditions which assure existence of these objects. Treatment of these conditions are usually complex and we need to have a mechanism to do that.

Chapter 3

Orientation and its applications

3.1 Introduction

Traditional geometry reasoning usually relies on tacit assumptions which are based on visual evidence. Even in a formal system such as Hilbert's one, many proofs are still based on reasoning with diagrams. We consider the following example to figure out how assumptions are tacitly interleaved in a traditional proof. Given a parallelogram ABCD with diagonals AC BD, we prove that they bisect each other.

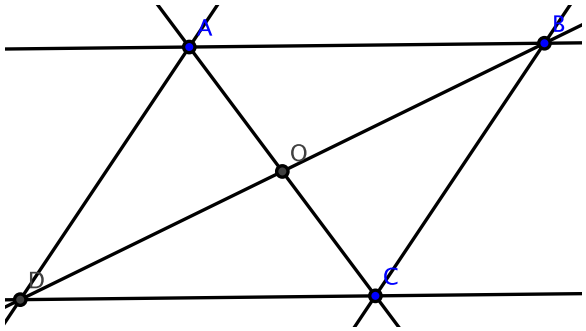


Figure 3.1: Parallelogram

Proof.

1. $AD \parallel BC$ (Since the opposite sides of a parallelogram are parallel)
2. $\overrightarrow{AD}\overrightarrow{AC} = \overrightarrow{CB}\overrightarrow{CA}$ (2 alternate interior angles of parallel lines are equal)
3. $\overrightarrow{AD}\overrightarrow{AO} = \overrightarrow{CB}\overrightarrow{CO}$ (From (2) and $\overrightarrow{AD}\overrightarrow{AO} = \overrightarrow{AD}\overrightarrow{AC}$ and $\overrightarrow{CB}\overrightarrow{CO} = \overrightarrow{CB}\overrightarrow{CA}$)
4. $\overrightarrow{DB}\overrightarrow{DA} = \overrightarrow{BD}\overrightarrow{BC}$ (2 alternate interior angles of parallel lines are equal)
5. $\overrightarrow{DO}\overrightarrow{DA} = \overrightarrow{BO}\overrightarrow{BC}$ (From (4) and $\overrightarrow{DO}\overrightarrow{DA} = \overrightarrow{DB}\overrightarrow{DA}$ and $\overrightarrow{BO}\overrightarrow{BC} = \overrightarrow{BD}\overrightarrow{BC}$)
6. AD = BC (Since the opposite sides of a parallelogram are equal)
7. $\triangle AOD \simeq \triangle COB$ (Two triangles are congruent by Side-Angle-Side (3)(5)(6))
8. So we have AO = CO and BO = DO (Corresponding sides of congruent triangles

are congruent).

The proof appears to be quite logical. But it still has some flaws. In fact, in proving the congruence of triangles in step 7, we use equality in step 3. The first argument to deduce this equality is $\widehat{\overrightarrow{AD}\overrightarrow{AC}} = \widehat{\overrightarrow{CB}\overrightarrow{CA}}$ that is affirmed in step 2. This fact seems to be evident because these angles are 2 alternate interior angles. However, this is true only if we have an assumption that A and C lie on opposite sides of the line BD.

The other arguments of step 3 are $\widehat{\overrightarrow{AD}\overrightarrow{AO}} = \widehat{\overrightarrow{AD}\overrightarrow{AC}}$ and $\widehat{\overrightarrow{CB}\overrightarrow{CO}} = \widehat{\overrightarrow{CB}\overrightarrow{CA}}$. To have these, we need an assumption that O is inside the segment AC.

One of the main defects in traditional proofs taught in high school geometry is its almost complete disregard of assumptions which concern such notions as two sides of a line, betweenness of two points, interior of an angle, etc. These rely on intuition and exactness of drawing, and are implicitly used without proving. This causes non-rigorous proofs. It even leads to wrong reasoning if the assumptions come from an incorrect drawing. Proving these conditions is tedious and sometimes more difficult than proving the original problems.

In fact, these conditions are related to a notion of order relation. Let's consider this notion in different geometry system.

Order relation with algebraic geometry: Our example, in an algebraic view, is a theorem of equality type. In fact, this class of theorems contains most of the important theorems of elementary geometry although it excludes theorems involving order relation. Some algebraic or semi-algebraic methods achieved success in mechanically proving theorems of this class. Algebraic methods such as Wu's method [9], the Grobner basis method [8] and semi-algebraic (so called coordinate free) methods such as the area method [11] and the full angle method [12] can solve problems without using any order relation.

However, these methods are restricted to theorems in the class which can be expressed by only equalities. For theorems in which the order relation is essential, proving them is beyond the scope of these methods.

Besides, algebraic methods are based on calculations of polynomials, their proofs are not readable. Coordinate free methods are based on geometric invariants which are more intuitive, then generated proofs are human-readable but still not fully traditional. Furthermore, these methods are automatic methods, hence they do not allow us to construct traditional proofs interactively.

Order relation with synthetic geometry: We realize that, although our example is stated in unordered geometry, the traditional proof is synthetically performed using ordered geometry. Order relations are implicitly used in the proofs. This shows that order relations are not only indispensable in order geometry theorems, but also necessary for synthetic proofs of some unordered geometry theorems.

In fact, the treatment of order relations is still a challenge in synthetic geometry. As we mentioned in the previous chapter, Euclidean's proofs are considered to be not rigorous because they disregard order relations. To fill this gap, axiom systems such as Hilbert's and Tarski's were proposed. Some axioms about order relation are introduced in these systems.

The above analysis motivates us to formalize the notion of order. In fact this notion was not present in F. Guilhot's library when we started our work. Formalizing order relation and constructing a tool to solve order problems were necessary. Our approach for this objective relies on constructing the notion of plane orientation. Instead of introducing axioms for order relation, we construct orientation for plane geometry in our library. We then use this to verify properties concerning the order relation. With this notion, we can also remove ambiguity in point positions or representation of geometric objects and we can formalize other notions such as oriented angles of vectors, oriented angles of lines, or similar triangles, which play important roles in many geometry theorems.

Related work

Some formalizations of orientation were developed in the Coq system. The first approach was proposed by D.Pichardie and Y.Bertot [14] based on the axiom system of D.Knuth [34] to solve convex hull problems. This approach uses an algebraic method with calculations of determinants to verify the axiom system. However, algebraic calculations do not preserve the intuition of geometry. The second approach was proposed by J.Duprat [18] for the purpose of reasoning on constructions of figures in plane geometry. In addition, this notion is also approached in the development of J.Avigad et al [2] which provide a formal system for Euclid's Elements. Except the first approach, the other ones consider orientation as an abstract notion and have axiom systems to manipulate it.

3.2 Definition of orientation

Consider three distinct and non-aligned points in Euclidean plane. They form a predicate about orientation. The notion of orientation is not absolute, but only relative. However, we can completely pick one orientation to say about this predicate. In particular, we here use the orientation of O, I and J to define orientation (denoted by ↺)

Let's consider a oriented points A, B, and C in the system of OIJ (in Fig 3.2). Let B' be the image of B after applying rotation with the center A and the angle $\overrightarrow{OI}\overrightarrow{OJ}$. It is easy to agree that OIJ and ABB' have the same orientation, so ABC is oriented if ABC and ABB' have the same orientation. This is expressed by B' and C lying on the same half plane with respect to the line AB. Therefore the orthogonal projection

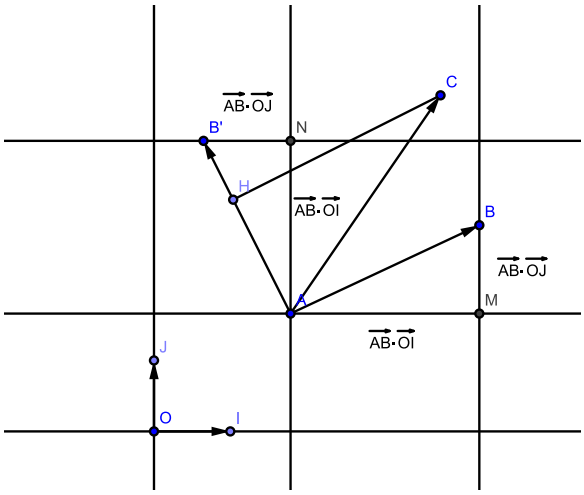


Figure 3.2: Constructing orthogonal vector

H of C on AB' is on the same side as B' with respect to A. In other words, we have $\cos \widehat{\overrightarrow{AC}\overrightarrow{AB'}} > 0$.

Since $\overrightarrow{AC} \cdot \overrightarrow{AB'} = |AC||AB'| \cos \widehat{\overrightarrow{AC}\overrightarrow{AB'}}$, it is easy to find that the above inequality is equivalent with $\overrightarrow{AC} \cdot \overrightarrow{AB'} > 0$. This new inequality can be used to define the orientation of ABC.

However, a question arising is how we can formally construct $\overrightarrow{AB'}$. As B' is the image of B after a rotation with center A and angle $\widehat{\overrightarrow{OI}\overrightarrow{OJ}}$, we easily find that $\overrightarrow{AB'}$ satisfies $|\overrightarrow{AB'}| = |\overrightarrow{AB}|$ and $\widehat{\overrightarrow{AB}\overrightarrow{AB'}} = \widehat{\overrightarrow{OI}\overrightarrow{OJ}}$. As a result, $\overrightarrow{AB'}$ is exactly orthogonal vector of $\overrightarrow{AB}$ that is defined in the previous chapter. Thus we can formally define orientation as follows:

Definition $\circlearrowleft$(A B C :Point):= $\overrightarrow{AB}^{\perp} \cdot \overrightarrow{AC} > 0$.

3.3 Some properties of orientation

3.3.1 The properties of Knuth's system

We now prove some properties of orientation. We pay attention to the ones stated in the axiom system of Knuth [34]. The following properties are equivalent with Knuth's axioms.

Property Orient 1. : The orientation is preserved with a cyclic permutation:

$$\circlearrowleft ABC \rightarrow \circlearrowleft BCA.$$

Property Orient 2. : The orientation is changed with a permutation, we don't have $\circlearrowleft ABC$ and $\circlearrowleft ACB$ at the same time:

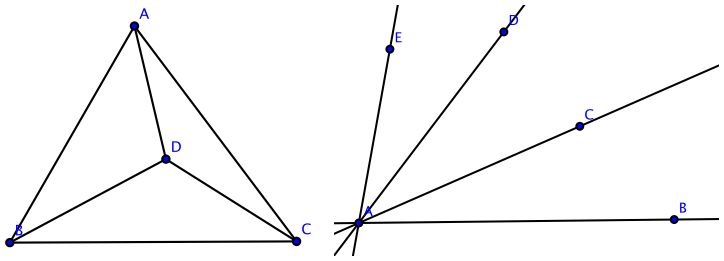
$$\circlearrowleft ABC \rightarrow \neg \circlearrowleft ACB.$$

Property Orient 3. : This property corresponds with to axiom 3 in Knuth's system stated as

$$\circlearrowleft ABC \vee \circlearrowleft ACB.$$

This system does not deal with the degenerated case. It assumes that every three points are never collinear. In our system, we distinguish the case where three points are collinear and the case where three points are oriented. Hence, for a triple of points A, B and C, we only fall into one of three cases $\circlearrowleft ABC$, $\circlearrowleft ACB$ or *collinear* ABC. This is represented by the following two lemmas.

Lemma position_3points_1 : ∀ (A B C :Point),
$\neg(\circlearrowleft ABC \vee \circlearrowleft ACB) \vee \neg collinear\ ABC$.
Lemma position_3points_2 : ∀ (A B C :Point),
$\circlearrowleft ABC \vee \circlearrowleft ACB \vee\ collinear\ ABC$.



(a) Orientation for 4 points (b) Orientation for 5 points

Figure 3.3

Property Orient 4. : If we have orientation of three triplets $\circlearrowleft DAB$ $\circlearrowleft DBC$ $\circlearrowleft DCA$, we can deduce that A, B, C are oriented: $\circlearrowleft DAB \wedge \circlearrowleft DBC \wedge \circlearrowleft DCA \rightarrow \circlearrowleft ABC$. This property is illustrated in Fig. 3.3(a).

Property Orient 5. : The orientation predicate may be used to sort points in some way, this is similar to transitivity: $\circlearrowleft ABC \wedge \circlearrowleft ABD \wedge \circlearrowleft ABE \wedge \circlearrowleft ACD \wedge \circlearrowleft ADE \rightarrow \circlearrowleft ACE$. This property is illustrated in Fig. 3.3(b).

3.3.2 Proving the properties

Proofs of these properties are performed by transformations and calculations of vectors, orthogonal vectors, and their scalar products. Properties of vector and scalar product are mentioned in the previous chapter. We list some properties used for transformations of orthogonal vector :

anti_Symmetry : $\overrightarrow{u}^{\perp} \cdot \overrightarrow{v} = -\overrightarrow{u} \cdot \overrightarrow{v}^{\perp}$.
ortho_Addition : $(\overrightarrow{u} + \overrightarrow{v})^{\perp} = \overrightarrow{u}^{\perp} + \overrightarrow{v}^{\perp}$.
ortho_Multiplication : $(k\overrightarrow{v})^{\perp} = k\overrightarrow{v}^{\perp}$.
ortho_Zero : $\overrightarrow{u}^{\perp} \cdot \overrightarrow{u} = 0$.

We now return to the properties of orientation. Proofs of the first three properties of orientation are straightforward, so we focus on proofs of properties 4 and 5.

Proof of property 4: $\forall ABCD, ⟳ DAB \wedge ⟳ DBC \wedge ⟳ DCA \rightarrow ⟳ ABC$

In mathematics, the proof of this property derives from a view about area of a triangle. Intuitively, a point D satisfying hypotheses in the statement has to lie inside $\triangle ABC$, hence

$$\mathcal{S}_{ABC} = \mathcal{S}_{DAB} + \mathcal{S}_{DBC} + \mathcal{S}_{DCA}.$$

Besides, let's see the definition of orientation $⟳ ABC := \overrightarrow{AB}^{\perp} \cdot \overrightarrow{AC} > 0$. In mathematic, the value of the left part of this inequality expresses the area of parallelogram composed by $\overrightarrow{AB}$ and $\overrightarrow{AC}$, hence this is equivalent with two times the area of $\triangle ABC$. This motivates us to prove that $\overrightarrow{DA}^{\perp} \cdot \overrightarrow{DB} + \overrightarrow{DB}^{\perp} \cdot \overrightarrow{DC} + \overrightarrow{DC}^{\perp} \cdot \overrightarrow{DA} = \overrightarrow{AB}^{\perp} \cdot \overrightarrow{AC}$. Once this is proved, we can deduce $⟳ DAB \wedge ⟳ DBC \wedge ⟳ DCA \rightarrow ⟳ ABC$ as follows $⟳ DAB \wedge ⟳ DBC \wedge ⟳ DCA \Rightarrow \overrightarrow{DA}^{\perp} \cdot \overrightarrow{DB} > 0 \wedge \overrightarrow{DB}^{\perp} \cdot \overrightarrow{DC} > 0 \wedge \overrightarrow{DC}^{\perp} \cdot \overrightarrow{DA} > 0 \Rightarrow \overrightarrow{DA}^{\perp} \cdot \overrightarrow{DB} + \overrightarrow{DB}^{\perp} \cdot \overrightarrow{DC} + \overrightarrow{DC}^{\perp} \cdot \overrightarrow{DA} > 0 + 0 + 0 \Rightarrow \overrightarrow{AB}^{\perp} \cdot \overrightarrow{AC} > 0 \Rightarrow ⟳ ABC$

The remaining work is to prove this equation. This is realized by a sequence of transformations of vectors and scalar products:

1. $\overrightarrow{AB}^{\perp} \cdot \overrightarrow{AC} = (\overrightarrow{DB} - \overrightarrow{DA})^{\perp} \cdot (\overrightarrow{DC} - \overrightarrow{DA})$ (replacement of vector)
2. $\overrightarrow{AB}^{\perp} \cdot \overrightarrow{AC} = (\overrightarrow{DB}^{\perp} - \overrightarrow{DA}^{\perp}) \cdot (\overrightarrow{DC} - \overrightarrow{DA})$ (application of ortho_Distributivity property)
3. $\overrightarrow{AB}^{\perp} \cdot \overrightarrow{AC} = \overrightarrow{DB}^{\perp} \cdot \overrightarrow{DC} - \overrightarrow{DB}^{\perp} \cdot \overrightarrow{DA} - \overrightarrow{DA}^{\perp} \cdot \overrightarrow{DC} + \overrightarrow{DA}^{\perp} \cdot \overrightarrow{DA}$(transformation of scalar product)
4. $\overrightarrow{AB}^{\perp} \cdot \overrightarrow{AC} = \overrightarrow{DB}^{\perp} \cdot \overrightarrow{DC} + \overrightarrow{DA}^{\perp} \cdot \overrightarrow{DB} + \overrightarrow{DC}^{\perp} \cdot \overrightarrow{DA} + 0$(transformation of scalar product)
5. $\overrightarrow{AB}^{\perp} \cdot \overrightarrow{AC} = \overrightarrow{DA}^{\perp} \cdot \overrightarrow{DB} + \overrightarrow{DB}^{\perp} \cdot \overrightarrow{DC} + \overrightarrow{DC}^{\perp} \cdot \overrightarrow{DA}$(transformation of scalar product) □.

.....*Begin of Technical Details*..... 3.

The proof of this property in Coq conforms to the mathematical proof. The replacements of vectors and transformations of scalar products are performed manually.

```
Lemma orient_4points: forall (A B C D : PO),
orient D A B -> orient D B C -> orient D C A -> orient A B C.
```

We introduce the hypotheses, and unfold every occurrence of orientation in the context by using its definition. We have:

```
Proof.
intros;unfold orient in *.

H0 : scalarProduct (orthoVect \overrightarrow{DA}) \overrightarrow{DB} > 0
H1 : scalarProduct (orthoVect \overrightarrow{DB}) \overrightarrow{DC} > 0
H2 : scalarProduct (orthoVect \overrightarrow{DC}) \overrightarrow{DA} > 0
------------------------------------(1/1)
scalarProduct (orthoVect \overrightarrow{AB}) \overrightarrow{AC} > 0
```

We replace $\overrightarrow{AB}$ by $(-1)\overrightarrow{DA}+\overrightarrow{DB}$, $\overrightarrow{AC}$ by $(-1)\overrightarrow{DA}+\overrightarrow{DC}$. These replacement is ensured by Chasles' relation for vectors and automatically solved by the tactic *RingVector* containing this relation.

We then perform transformations of orthogonal vectors to simplify the proof obligation. In particular, $((-1)\overrightarrow{DA}+\overrightarrow{DB})^{\perp}$ is transformed to $(-1)\overrightarrow{DA}^{\perp}+\overrightarrow{DB}^{\perp}$. This uses the properties *ortho_Addition* and *ortho_Multiplication*.

These steps are corresponding to steps 1-2 of the above mathematic proof and give us:

```
...
replace (\overrightarrow{AB}) with ((-1)\overrightarrow{DA} + \overrightarrow{DB}) by RingVector.
replace (\overrightarrow{AC}) with ((-1)\overrightarrow{DA} + \overrightarrow{DC}) by RingVector.
rewrite ortho_Addition .
rewrite orthoVect_Multiplication.

H0: ...
------------------------------------(1/1)
scalarProduct ((-1)(orthoVect \overrightarrow{DA}) +  orthoVect \overrightarrow{DB})
              ((-1)(\overrightarrow{DA}) + \overrightarrow{DC}) > 0
```

We now use the tactic *RingScalarProduct*(mentioned on page 30) to simplify the formula in the proof obligation and have:

```
...
RingScalarProduct.

H0:...
------------------------------------(1/1)
scalarProduct (orthoVect \overrightarrow{DA}) \overrightarrow{DA} -
scalarProduct (orthoVect \overrightarrow{DA}) \overrightarrow{DC} -
scalarProduct (orthoVect \overrightarrow{DB}) \overrightarrow{DA} +
scalarProduct (orthoVect \overrightarrow{DB}) \overrightarrow{DC} > 0
```

We use the property *ortho_Zero* to replace $\overrightarrow{DA}^{\perp}\cdot\overrightarrow{DA}$ with 0. We use the property *anti_Symmetry* to replace $-\overrightarrow{DA}^{\perp}\cdot\overrightarrow{DC}$ with $(-\ -\overrightarrow{DA}\cdot\overrightarrow{DC}^{\perp})$, then the symmetry

of scalar product to replace $(- \ - \overrightarrow{DA} \cdot \overrightarrow{DC}^{\perp})$ with $\overrightarrow{DC}^{\perp} \cdot \overrightarrow{DA}$. In a similar way, $-\overrightarrow{DA}^{\perp} \cdot \overrightarrow{DC}$ is replaced with $\overrightarrow{DA}^{\perp} \cdot \overrightarrow{DB}$. We have:

```
...
rewrite ortho_Zero.
rewrite  (@anti_Symmetry DA DC).
rewrite (@scalarProduct_Symmetry (orthoVect (DA)) DC)
rewrite  (@anti_Symmetry DB DA).
rewrite (@scalarProduct_Symmetry (orthoVect DB) DC)
ring_simplify.

H0 : scalarProduct (orthoVect DA) DB > 0
H1 : scalarProduct (orthoVect DB) DC > 0
H2 : scalarProduct (orthoVect DC) DA > 0
------------------------------------(1/1)
scalarProduct (orthoVect DC) DA +
scalarProduct (orthoVect DA) DB +
scalarProduct (orthoVect DB) DC > 0
```

This proof obligation is easily proved using the tactic *fourier*. This tactic allows us to solve linear inequalities on real numbers using Fourier's method.

```
...
fourier.

Proof completed.
```

.....End of Technical Details.....

Proof of property 5: $\forall ABCDE, \circlearrowleft ABC \wedge \circlearrowleft ABD \wedge \circlearrowleft ABE \wedge \circlearrowleft ACD \wedge \circlearrowleft ADE \rightarrow \circlearrowleft ACE$

This property is proved as in [14]. However, we use here a variant of Cramer's equation, which is stated in the scalar product form as follows

$\overrightarrow{AB}^{\perp} \cdot \overrightarrow{AD} \times \overrightarrow{AC}^{\perp} \cdot \overrightarrow{AE} = \overrightarrow{AB}^{\perp} \cdot \overrightarrow{AC} \times \overrightarrow{AD}^{\perp} \cdot \overrightarrow{AE} + \overrightarrow{AC}^{\perp} \cdot \overrightarrow{AD} \times \overrightarrow{AB}^{\perp} \cdot \overrightarrow{AE}$.

The hypotheses about orientation give us

$$\overrightarrow{AB}^{\perp} \cdot \overrightarrow{AC} > 0 \wedge \times \overrightarrow{AB}^{\perp} \cdot \overrightarrow{AD} > 0 \wedge \overrightarrow{AB}^{\perp} \cdot \overrightarrow{AE} > 0 \wedge \overrightarrow{AC}^{\perp} \cdot \overrightarrow{AD} > 0 \wedge \overrightarrow{AD}^{\perp} \cdot \overrightarrow{AE} > 0.$$

It is not difficult to see that every scalar products appearing in the right part of the equality is positive, hence the right part of the equality is also positive. Besides, the first scalar product of the left part of the equality is positive. Therefore we can readily prove the positivity of the second element of this part ($\overrightarrow{AC}^{\perp} \cdot \overrightarrow{AE} > 0$), hence $\circlearrowleft ACE$.

Due to the complexity of this equality, the proof technique that uses manual replacement of vectors as in the proof of property 4 is difficult to apply. Fortunately, using representation of vectors in the OIJ system ($\overrightarrow{v} = (\overrightarrow{v} \cdot \overrightarrow{OI})\overrightarrow{OI} + (\overrightarrow{v} \cdot \overrightarrow{OJ})\overrightarrow{OJ}$)

allows us to perform calculations of scalar products through their coordinate values. This allows us to prove this equality.

.....Begin of Technical Details..... 4.
The following proof in Coq of this property show the capability of our library in calculations of scalar products using their coordinate values. This power is useful in many problems where manual transformation is not easily performed.

The proof context is as follows

```
______________________________________(1/1)
scalarProduct (orthoVect AB) AD *
scalarProduct (orthoVect AC) AE =
scalarProduct (orthoVect AB) AC *
scalarProduct (orthoVect AD) AE +
scalarProduct (orthoVect AB) AE *
scalarProduct (orthoVect AC) AD
```

For every vector in the proof obligation, we replace $\overrightarrow{v}$ by $(\overrightarrow{v} \cdot \overrightarrow{OI})\overrightarrow{OI} + (\overrightarrow{v} \cdot \overrightarrow{OJ})\overrightarrow{OJ}$ and $\overrightarrow{v}^{\perp}$ by $-(\overrightarrow{v} \cdot \overrightarrow{OJ})\overrightarrow{OI} + (\overrightarrow{v} \cdot \overrightarrow{OI})\overrightarrow{OJ}$(using the definition of orthogonal vector). We then use x_v, y_v to respectively denotes $(\overrightarrow{v} \cdot \overrightarrow{OI})$ and $(\overrightarrow{v} \cdot \overrightarrow{OJ})$. These lead us to the following proof obligation:

```
repeat unfold orthoVect.
RingScalarProduct.
set (XB := scalarProduct AB OI).
replace (scalarProduct AB OI) with XB
         by (unfold XB; auto with geo).
set (YB := scalarProduct AB OJ).
replace (scalarProduct AB OJ) with YB
         by (unfold YB; auto with geo).

______________________________________(1/1)
YB * XD * YC * XE - YB * XD * XC * YE -
 XB * YD * YC * XE + XB * YD * XC * YE =
YB * XD * YC * XE - YB * YD * XE * XC +
YB * XE * XC * YD - YB * YE * XC * XD -
XD * XB * YC * YE + XB * YD * XC * YE -
XB * XE * YC * YD + XB * YE * YC * XD
```

This proof obligation seems to be very complex. However, this is solved with only one command using the tactic *ring*:

```
...
ring.

Proof completed.
```

.....End of Technical Details.....

3.3.3 Variants

We note that, as mentioned, the notion of orientation is a relative, we pick one direction (counter-clockwise) to define this notion. Therefore, every propertiy about orientation can be generalized with the notion of same orientation (denoted by $\circlearrowleft\circlearrowleft\ ABCMNP$). For example, property 4 can be stated in a generalized form as

$\circlearrowleft\circlearrowleft\ DABDBC \wedge \circlearrowleft\circlearrowleft\ DABDCA \rightarrow \circlearrowleft\circlearrowleft\ DABABC$

```
Lemma sameOrient_4points: forall (A B C D : PO),
sameOrient D A B D B C-> sameOrient D A B D C A ->
sameOrient D A B A B C.
```

Where the notion of same orientation is defined by

```
Definition sameOrient A B C M N P  :=
```
$(\circlearrowleft\ ABC \wedge \circlearrowleft\ MNP) \vee (\circlearrowleft\ ACB \wedge \circlearrowleft\ MPN)$.

This notion makes theorem statements more concise. It allows us to avoid stating theorems in the two cases corresponding to the two orientation. However, proving theorems stated with *samOrient* is performed by considering both cases of orientation.

3.4 Orientation and order

3.4.1 Properties

Orientation and order are closely related. They both express relative positions among geometric objects (which usually are points). Enriching our system with orientation allows us to define some order relations and verify their properties. Before explaining what we can do, we need to define some basic order relations that we usually meet in solving order geometry problem. They include: same side of point, between 2 points, same side or opposite side of line, interior of angle ...

same side: we say that B and C lie on the same side to A (denoted by $\overrightarrow{ABC}$) if they are 3 distinct points that lie on the same line and two vectors $\overrightarrow{AB}$ and $\overrightarrow{AC}$ are positive collinear. Intuitively, this notion says that C lies on the half line from A to B and differs from A.

```
Definition sameSide A B C :=
```
$A <> B \wedge (\exists k, k > 0 \wedge \overrightarrow{AC} = k\overrightarrow{AB})$.

between: the fact that B lies between A and C (denoted by $\overline{ABC}$) is expressed by B and C lying on the same side of A and A and B lying on the same side of C.

```
Definition between A B C := sameSide A B C ∧ sameSide C B A.
```

same side of line: 2 points C and D lie on the same side of the line AB if ABC and ABD have the same orientation

```
Definition sameSide_ofLine A B C D := sameOrient A B C A B D.
```

interior: we say that D is interior of angle formed by B, A and C (in other words, $\overrightarrow{AD}$ lies between $\overrightarrow{AB}$ and $\overrightarrow{AC}$) if and only if D and C lie on the same side of line AB and D and B lie on the same side of line AC. With the definition of same orientation, this notion can be expressed by

```
Definition interior A B C D :=sameOrient ABC ABD ∧ sameOrient ABC ADC.
```

To facilitate explantation, we use ↻↻ $ABCMNP$ to denote same orientation of ABC and MNP, $\overline{ABC}$ to denote that B is between A and C, $\overrightarrow{ABC}$ to denote that B and C lie on the same side of A

We now consider some interesting results concerning relations between orientation and the newly defined notions. We always focus on the relative position of a fourth point with respect to three given points. The following two properties are simple cases when the fourth point D lies on the line AB. Similarly, we have corresponding properties for the case where D lies on the line AC but we do not give details here.

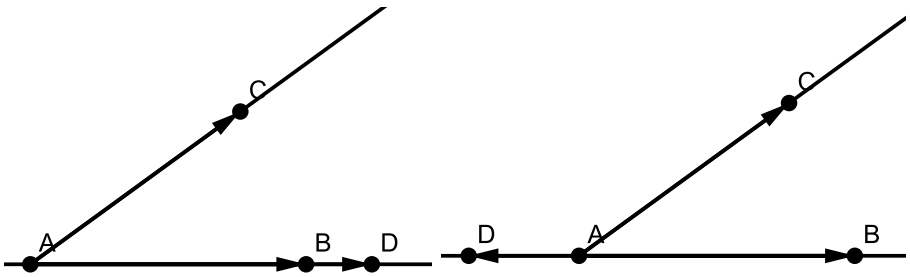


Figure 3.4: Relations between the same side and the same orientation

Property Orient 6. : Given 4 points A, B, C and D such that A, B and C are not collinear and D lies on the line AB and differs from A. Then ABC and ADC have the same orientation if and only if D lies on the same side of A as B (see Fig. 3.4(a)).

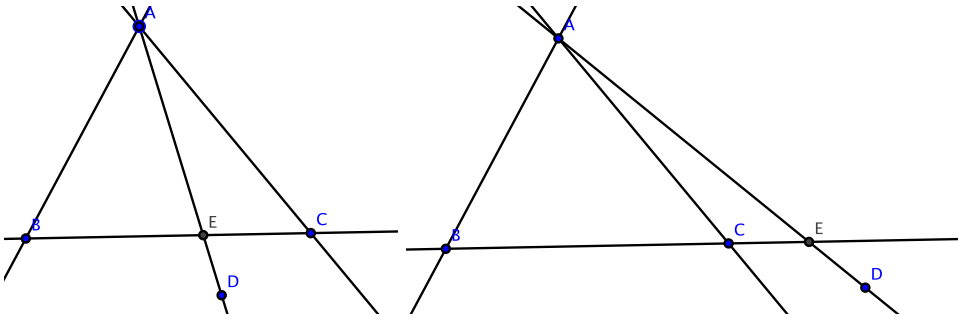
```
Lemma sameSide_sameOrient_left1 :=
```
$\forall ABCD, \neg col\ ABC \rightarrow sameSide\ A\ B\ D \rightarrow sameOrient\ A\ B\ C\ A\ D\ C\,.$
```
Lemma sameSide_sameOrient_left2 :=
```
$\forall ABCD, \neg col\ ABC \rightarrow D \in AB \rightarrow sameOrient\ A\ B\ C\ A\ D\ C \rightarrow sameSide\ A\ B\ D\,.$

Property Orient 7. : Given 4 points A, B, C and D such that A, B and C are not collinear and D lies on the line AB and differs from A. Then ABC and ACD have the same orientation if and only if A is between D and B (see Fig. 3.4(b)).

```
Lemma between_sameOrient_left1 :=
```
$\forall ABCD, \neg col\ ABC \rightarrow between\ D\ A\ B \rightarrow sameOrient\ A\ B\ C\ A\ C\ D\,.$

Lemma between_sameOrient_left2 :=
$\forall ABCD, \neg col\ ABC \rightarrow D \in AB \rightarrow sameOrient\ A\ B\ C\ A\ C\ D \rightarrow between\ D\ A\ B$.

We focus now on more complex relations where D does not lie either on AB or on AC. That is the case of the interior relation. We consider configurations of points in Fig. 3.5 and are interested in determining the relative position of the intersection point E of lines AD and BC with respect to A, B, C and D. The first case is illustrated by Fig. 3.5(a) which corresponds to the case where $\overrightarrow{AD}$ lies between $\overrightarrow{AB}$ and $\overrightarrow{AC}$, and will be treated in the following property 8. The second case is illustrated by Fig. 3.5(b), which correponds to the case where $\overrightarrow{AC}$ lies between $\overrightarrow{AB}$ and $\overrightarrow{AD}$, and will be treated in the following property 9.



(a) (b)

Figure 3.5: Intersection point

Property Orient 8. : Given four points A, B, C and D, if $\overrightarrow{AD}$ is between $\overrightarrow{AB}$ and $\overrightarrow{AC}$, then AD and BC always intersect and the intersection point E is in the segment BC and on the same side as D with respect to A. It's still true in the inverse direction(see Fig. 3.5(a)).

Lemma Intersect_interior : $\forall(A\ B\ C\ D : Point), interior\ A\ B\ C\ D \rightarrow \exists E : Point,\ between\ B\ E\ C \wedge sameSide\ A\ D\ E$.

Property Orient 9. : Given four points A, B, C and D, if $\overrightarrow{AC}$ is between $\overrightarrow{AB}$ and $\overrightarrow{AD}$ and AD and BC intersect at the point E, then E lies on the same side as D with respect to A and E lies on the same side as C with respect to B (see Fig. 3.5(b)).

Lemma Intersect_exterior : ∀ (A B C D E:Point),
interior A B D C ∧ collinear BCE ∧ collinear ADE →
(sameSide B C E ∧ sameSide A D E).

These properties are used a lot when verifying order axioms in the axiom systems of Hilbert and Tarski. In particular, they are used in proving Pasch' axiom and its variant. This will be detailed in the next chapter.

3.4.2 Proving the properties

In this section, we only consider the hardest proof that is for property 8. Proving this theorem contains 3 steps. In the first step, we prove the existence of intersection point of lines AD and BC. This is performed by proving $AD \nparallel BC$. The other two steps are to prove respectively $\overrightarrow{ADE}$ and $\overline{BEC}$. In the following explanation of the proofs, hypotheses of distinction of points such as $E \neq B$, $E \neq A$, $E \neq C$ are used without proving. Their proofs are not hard, but long and tedious.

For proving $\overrightarrow{ADE}$, we use proof by cases over the position of E with respect to A and D, we have 3 cases $\overline{ADE} \vee \overline{DEA} \vee \overline{EAD}$. For the first two cases, it is easy to prove $\overrightarrow{ADE}$. We consider the last case, where we have $\overline{EAD}$. Applying property 7 with 4 points A D C E, we have $\overline{EAD} \leftrightarrow ↻↻\ ADCACE$, hence $↻↻\ ADCACE$. On the other hand, from the hypothesis *interior* $ABCD$, by the definition of this notion, we have $↻↻\ ABCADC$.

$↻↻\ ADCACE$ and $↻↻\ ABCADC$ allow us to deduce $↻↻\ ABCACE$. By a similar proof, we get $↻↻\ ABCAEB$.

We see that the configuration of E, B, C and A satisfies the variant of property 4 in the last section (see page 56). Applying this property with the configuration gives $↻↻\ ABCAEB \wedge\ ↻↻\ ABCACE \rightarrow ↻↻\ ABCEBC$, hence $↻↻\ ABCEBC$. This leads to a contradiction with the collinearity of B, C and E.

For proving $\overline{BEC}$, we also use proof by cases over the position of E, with 3 cases $\overline{BEC} \vee \overline{EBC} \vee \overline{ECB}$. For the first case, our proof is finished since we have $\overline{BEC}$. The latter are equivalent by switching B and C, so they are treated by the same way.

We now continue to prove with the case of $\overline{EBC}$. Applying property 7 with 4 points B, C, A and E, we have $\overline{EBC} \leftrightarrow ↻↻\ BCABAE$, hence $↻↻\ BCABAE$. This is equivalent with $↻↻\ ABCAEB$.

From the hypothesis *interior* $ABCD$, by the definition of this notion, we have $↻↻\ ABCABD$. Besides, we already proved $\overrightarrow{ADE}$. From the last two, we have $↻↻\ ABDABE$, hence $↻↻\ ABCABE$.

So we have $↻↻\ ABCAEB$ and $↻↻\ ABCABE$ in the same time. This leads to $↻↻\ ABEAEB$. This is contradictory with the property 2 which say that if we have $↻\ ABE$ we do not have $↻\ AEB$. Thus, our proof of this case is finished. □

.....*Begin of Technical Details*..... 5.

In comparing with the mathematic proof mentioned above, the proof of this property in Coq is very long with more than 100 proof lines. We have to prove many degenerate cases and auxiliary properties, for example: existence of E such that $E \in AD \wedge E \in BC$, difference of E with A, B, C and D, etc.

We here present only the part of the proof for $\overrightarrow{ADE}$. The proof context is as follows:

...

```
H0 : liesOnLine E (line A D)
H1 : liesOnLine E (line B C)
H12 : sameOrient A B C A B D
H14 : sameOrient A B C A D C
______________________________________(1/1)
sameSide A D E
```

From the hypothesis H0, we have the collinearity of A, D and E. A lemma, namely *between_3cases*, to state that we have one of 3 cases $\overline{ADE} \vee \overline{DEA} \vee \overline{EAD}$. The tactic *destruct* allows us to continue our proof with each case. The first two are simply proved using the following equivalences $\overline{ADE} \leftrightarrow \overrightarrow{ADE} \wedge \overrightarrow{EDA}$ and $\overline{DEA} \leftrightarrow \overrightarrow{DEA} \wedge \overrightarrow{ADE}$. It remains to prove the last case.

```
...
destruct (@between_3cases A D E) as [H17|[H17|H17]];auto.
rewrite (@between_sameSide A D E) in H17;
 destruct H17;auto with geo. (* for case 1*)
rewrite (@between_sameSide D E A) in H17;
 destruct H17;auto with geo. (* for case 2*)

H1 : liesOnLine E (line B C)
H12 : sameOrient A B C A B D
H14 : sameOrient A B C A D C
H17 : between E A D
______________________________________(1/1)
sameSide A D E
```

Applying property 7, particularly the lemma *between_sameOrient_left1*, with 4 points A, D, C, E and $\overline{EAD}$, we have $\circlearrowleft\circlearrowleft$ $ADCACE$. It is similar for $\circlearrowleft\circlearrowleft$ $ABDAEB$. We have

```
...
assert (sameOrient A B D A E B) by
 (apply between_sameOrient_left1;auto with geo).
assert (sameOrient A D C A C E ) by
 (apply between_sameOrient_left1;auto with geo).

H1 : liesOnLine E (line B C)
H12 : sameOrient A B C A B D
H14 : sameOrient A B C A D C
H18 : sameOrient A B D A E B
H19 : sameOrient A D C A C E
______________________________________(1/1)
sameSide A D E
```

From H12 and H18 we have $\circlearrowleft\circlearrowleft$ $ABCAEB$ and from H14 and H19 we have $\circlearrowleft\circlearrowleft$

$ABCACE$. These rely on the transitivity property of orient and are automatically solved. As a result, the configuration of B, C, E and A satisfies the extension of property 4 for same orientation (see page 56)

$$\circlearrowleft\circlearrowleft ABCACE \wedge \circlearrowleft\circlearrowleft ABCAEB \rightarrow \circlearrowleft\circlearrowleft ABCCEB.$$

Therefore we apply this property and have:

```
...
assert (sameOrient A B C C E B) by
 (apply sameOrient_4points;auto with geo).

H1 : liesOnLine E (line B C)
H12 : sameOrient A B C A B D
H14 : sameOrient A B C A D C
H18 : sameOrient A B D A E B
H19 : sameOrient A D C A C E
H20 : sameOrient A B C C E B
______________________________________(1/1)
sameSide A D E
```

Here, we unroll *sameOrient A B C C E B* to get $\neg col\ C\ E\ B$. This conflicts with the hypothesis H1 that gives us *col C E B*. This contradiction in the hypotheses allows us to prove the problem. This is performed by the tactic *intuition*.

```
...
assert (col C E B) by auto with geo.
assert (~col C E B) by unroll H20.
intuition.

Proof completed.
```

.....End of Technical Details.....

3.5 Some application

3.5.1 Orientation in formalizing some geometric notions

3.5.1.1 Orientation and Oriented Angle

Oriented angles of vectors (also called oriented angles or angles for abbreviation) and orientation are interlaced. Given three points A, B and C, $\overset{\frown}{\overrightarrow{AB}\overrightarrow{AC}}$ is defined by the record of $((\frac{\overrightarrow{AB}\cdot\overrightarrow{AC}}{|\overrightarrow{AB}|\times|\overrightarrow{AC}|}), (\frac{\overrightarrow{AB}^{\perp}\cdot\overrightarrow{AC}}{|\overrightarrow{AB}^{\perp}|\times|\overrightarrow{AC}|}))$. The sine function of this angle is defined by the second element $\sin\overset{\frown}{\overrightarrow{AB}\overrightarrow{AC}} = \frac{\overrightarrow{AB}^{\perp}\cdot\overrightarrow{AC}}{|\overrightarrow{AB}^{\perp}|\times|\overrightarrow{AC}|}$. It is obvious that the denominator of the fraction is positive ($|\overrightarrow{AB}^{\perp}| \times |\overrightarrow{AC}| > 0$) in the case $A \neq B \wedge A \neq C$ and the numerator

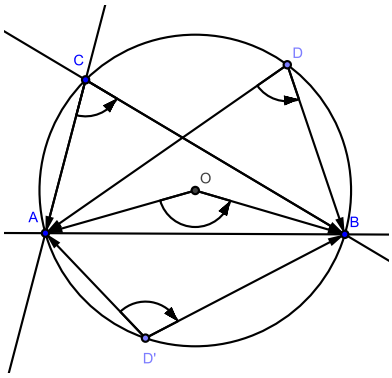


Figure 3.6: Inscribed angles

is used in defining orientation. As a consequence of the definition of orientation, we can deduce that if A, B and C are oriented then they are not collinear, hence they are distinct. We so easily get that $\circlearrowleft\ ABC \leftrightarrow A \neq B \wedge A \neq C \wedge \sin \overset{\frown}{\overrightarrow{AB}\overrightarrow{AC}} > 0$. This equivalence shows that our definition of orientation is consistent with the conventional approach that uses a positive value of the sine function to define orientation.

We now consider using orientation to remove ambiguity of representation of inscribed angles in the figure 3.6. The angles $\angle ACB$ and $\angle ADB$ are either equal or complementary according to the relative positions of C and D to the line AB (on the same side or opposite sides of the line AB). In usual proofs of geometry, this kind of ambiguity is treated by intuition, hence it depends on the exactness of drawing. However, with orientation, we can distinguish these cases. In fact, C and D are on the same side of the line AB expresses the same orientation of ACB and ADB. The fact that they are not on the same side expresses that there is a different orientation for ACB and ADB. So, deciding that $\angle ACB$ and $\angle ADB$ are either equal or complementary is formally represented and verified in the form of the following lemmas.

Property Orient 10. : two inscribed angles intercepting the same arc are equal if they have the same orientation.

Lemma inscribedAngles_equal: ∀ A B C D: Point,
$co\_cyclic\ A\ B\ C\ D \rightarrow \circlearrowleft\circlearrowleft\ CABDAB \rightarrow \overset{\frown}{\overrightarrow{CA}\overrightarrow{CB}} = \overset{\frown}{\overrightarrow{DA}\overrightarrow{DB}}$.

Property Orient 11. : two inscribed angles intercepting the same arc are complementary if they do not have the same orientation.

Lemma inscribedAngles_compl: ∀ A B C D': ␣Point,
$␣co\_cyclic\ A\ B\ C\ D' \rightarrow \circlearrowleft\circlearrowleft\ CABD'BA \rightarrow \overset{\frown}{\overrightarrow{CA}\overrightarrow{CB}} = \overset{\frown}{\overrightarrow{D'A}\overrightarrow{D'B}} + (-\pi)$.

To prove these properties, we first prove a lemma stating that the inscribed angle equals a half of central angle that intercepts the same arc. Given A, B and M which

are distinct points in the circle with the center O, this lemma is stated by $\overrightarrow{OA}\widehat{\overrightarrow{OB}} = \overrightarrow{MA}\widehat{\overrightarrow{MB}} +_{angle} \overrightarrow{MA}\widehat{\overrightarrow{MB}}$. This is readily proved by using Chasles lemma for oriented angles, the equivalence of base angles of an isosceles triangle and the sum of angles of a triangle.

Let's return to proofs of properties 10 and 11. We use α and β to respectively denote $\overrightarrow{CA}\widehat{\overrightarrow{CB}}$ and $\overrightarrow{DA}\widehat{\overrightarrow{DB}}$. Applying the above lemma with the triples of ABC and ABD gives us $\alpha +_{angle} \alpha = \beta +_{angle} \beta = \overrightarrow{OA}\widehat{\overrightarrow{OB}}$. By definition of the addition operator, we have

$(\sin\alpha \times \cos\alpha + \cos\alpha \times \sin\alpha, \cos\alpha \times \cos\alpha - \sin\alpha \times \sin\alpha) = (\sin\beta \times \cos\beta + \cos\beta \times \sin\beta, \cos\beta \times \cos\beta - \sin\beta \times \sin\beta)$.

Thus we have equalities of the corresponding elements as follows

$$\sin\alpha \times \cos\alpha = \sin\beta \times \cos\beta \tag{3.1}$$

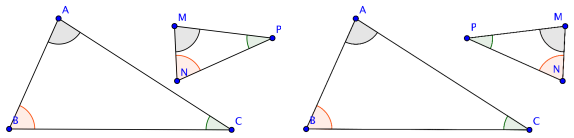
$$\sin\alpha \times \sin\alpha = \sin\beta \times \sin\beta \tag{3.2}$$

To prove property 10, from its hypothesis $\circlearrowleft\circlearrowleft\ CABDAB$, we have $\circlearrowleft\ CAB \wedge \circlearrowleft\ DAB$ or $\circlearrowleft\ CBA \wedge \circlearrowleft\ DBA$. The proofs for the 2 cases are similar, so we only consider the first case. We have $\circlearrowleft\ CAB$ and $\circlearrowleft\ DAB$. At the beginning of this section, we showed that $\circlearrowleft\ CAB \rightarrow \sin \overrightarrow{CA}\widehat{\overrightarrow{CB}} > 0$, hence $\sin\alpha > 0$. Similarly we have $\sin\beta > 0$. On the other hand we have $\sin\alpha \times \sin\alpha = \sin\beta \times \sin\beta$ from (3.2), so we get $\sin\alpha = \sin\beta$. By substituting $\sin\alpha$ for $\sin\beta$ in (3.1) , we get $\cos\alpha = \cos\beta$. The fact that $\sin\alpha = \sin\beta$ and $\cos\alpha = \cos\beta$ allows us to conclude $\alpha = \beta$ $\square$.

To prove property 11, in the same way, we consider the case that $\circlearrowleft\ CAB$ and $\circlearrowleft\ DBA$. We have $\sin\alpha > 0 \wedge \sin\beta < 0$, then $\sin\alpha = -\sin\beta$ and $\cos\alpha = -\cos\beta$. So we have $(\sin\alpha, \cos\alpha) = (-\sin\beta, -\cos\beta) = (\sin\beta \times (-1) + \cos\beta \times 0, \cos\beta \times (-1) - \sin\beta \times 0) = (\sin\beta \times \cos(-\pi) + \cos\beta \times \sin(-\pi), \cos\beta \times \cos(-\pi) - \sin\beta \times \sin(-\pi))$. By definition of oriented angle and addition operator we get $\alpha = \beta +_{angle} (-\pi)$ $\square$.

3.5.1.2 Orientation and Similar Triangles

Two triangles are similar if they have the same shape, but can be have different sizes. Learning the properties of similar triangles helps to solve problems about relationships in geometric figures and finding unknown quantities. Different approaches can be used to define it such as Angle-Angle (two pair of corresponding angles of triangles are equal), or Side-Side-Side (three pairs of corresponding sides of triangles are in proportion),etc . However, it is a fact that each of these conditions implies the others. We chose the first to define similar triangles (denoted by $\sim$). We divide this notion into direct and inverse similarity (see Fig 3.7). Two triangles are said to be directly similar (denoted by $\sim_d$) when all corresponding angles are equal and described in the same rotational



(a) Directly similar triangles (b) Inversely similar triangles

Figure 3.7: Directly similar triangles and Inversely similar triangles

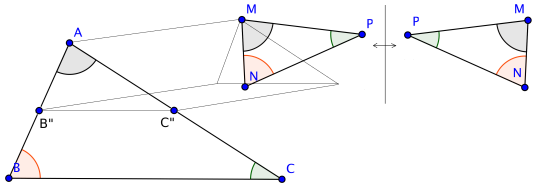


Figure 3.8: Demonstration of proportion

sense. They are said to be inversely similar (denoted by $\sim_i$) when they are equal but described in the opposite rotational sense. Their definitions are as follows

$$ABC \sim_d A'B'C' := \triangle ABC \bigwedge \triangle A'B'C' \bigwedge \widehat{\overrightarrow{BC}\overrightarrow{BA}} = \widehat{\overrightarrow{B'C'}\overrightarrow{B'A'}} \bigwedge \widehat{\overrightarrow{CA}\overrightarrow{CB}} = \widehat{\overrightarrow{C'A'}\overrightarrow{C'B'}}.$$
$$ABC \sim_i A'B'C' := \triangle ABC \bigwedge \triangle A'B'C' \bigwedge \widehat{\overrightarrow{BC}\overrightarrow{BA}} = \widehat{\overrightarrow{B'A'}\overrightarrow{B'C'}} \bigwedge \widehat{\overrightarrow{CA}\overrightarrow{CB}} = \widehat{\overrightarrow{C'B'}\overrightarrow{C'A'}}.$$
$$ABC \sim A'B'C' := ABC \sim_d A'B'C' \bigvee ABC \sim_i A'B'C'.$$

One of the most important properties is the proportionality of corresponding sides of two similar triangles. Given 2 triangles $\triangle ABC \sim \triangle MNP$, this property is stated by equation

$$\frac{AB}{MN} = \frac{BC}{NP} = \frac{CA}{PM} \tag{3.3}$$

Its traditional proof is readily performed using Euclidean transformations and Thales' theorem for parallel lines, illustrated in Fig. 3.8.

The case of inverse similarity is reduced to the one of direct similarity using reflection. Triangle $\triangle MNP$ is moved to triangle $\triangle AB"C"$ by applying a rotation with the angle $\widehat{\overrightarrow{MN}\overrightarrow{AB}}$ to have $MN \parallel AB$ and a translation to have $M \equiv A$.

These transformation lead to $\widehat{\overrightarrow{MN}\overrightarrow{AB"}} = \widehat{\overrightarrow{NM}\overrightarrow{AB}}$, hence collinearity of A, B" and B.

From the preservation of angle measures and lengths of Euclidean transformations, we can also prove that A, C" and C are collinear, $B"C" \parallel BC$, and $|AB"| = |MN| \wedge |AC"| = |MP| \wedge |B"C"| = |NP|$.

We observe that the configuration of A, B, C , B" and C" satisfies a consequence of Thales' theorem for parallel lines that is

$\forall ABCB"C", col\ A\ B\ B" \rightarrow col\ A\ C\ C" \rightarrow B"C" \parallel BC \rightarrow \frac{AB}{AB"} = \frac{AC}{AC"} = \frac{BC}{B"C"}$

Applying this give us $\frac{AB}{AB"} = \frac{AC}{AC"} = \frac{BC}{B"C"}$. By replacing the sides of $\triangle AB"C"$ by the corresponding ones of $\triangle MNP$, we have $\frac{AB}{MN} = \frac{BC}{NP} = \frac{CA}{PM}$. □

.....*Begin of Technical Details*..... 6.

We can now consider the proof of this property for direct similar case in Coq. The case is stated as follows

```
Lemma SimTriangles_Proportion :
 forall A B C M N P :Point, directSimilar A B C M N P ->
exists k:R, k <> 0 /\ distance B C = k * distance N P /\
distance A B = k * distance M N /\distance A C = k * distance M P .

H1 : directSimilar A B C M N P
------------------------------------(1/1)
exists k:R, k <> 0 /\ distance B C = k*distance N P /\
distance A B = k*distance M N /\ distance A C = k*distance M P .
```

In the first step, moving $\triangle MNP$ to $\triangle AB"C"$ is performed by defining two points $B"$ and $C"$ using constructive definitions of Euclidean transformations. In particular, these point are constructed by

```
set (B':= rotation M (angleCons (vector MN)(vector AB)) N).
set (B":=  translation (vector MA) B').
set (C':= rotation M (angleCons (vector MN)(vector AB)) P)
set (C":=  translation (vector MA) C').

H1 : directSimilar A B C M N P
B':= rotation M (angleCons (vector MN)(vector AB)) N.
B":=  translation (vector MA) B'.
C':= rotation M (angleCons (vector MN)(vector AB)) P
C":=  translation (vector MA) C'.
------------------------------------(1/1)
exists k:R, k <> 0 /\ distance B C =k*distance N P /\
distance A B = k*distance M N /\ distance A C = k*distance M P .
```

To introduce equalities of corresponding side of $\triangle MNP$ and $\triangle AB"C"$, we use preservations of length of rotation and translation as follows.

```
Lemma rotation_isometrie : forall (I A B A' B' : Point) (a : AV),
 A' = rotation I a A -> B' = rotation I a B ->
   distance A' B' = distance A B.
Lemma translation_isometrie : forall (A B A' B' : Point),
 B' = translation A A' B -> distance A' B' = distance A B.
```

Since B' is obtained from N with a rotation of the center M and the angle $\widehat{\overrightarrow{MN}\overrightarrow{AB}}$, using *rotation_isometrie* give us $|MN| = |MB'|$. In addition, B" is obtained from B with a translation of $\overrightarrow{MA}$, using *translation_isometrie* give us $|MB'| = |AB"|$. As a result, we get $|AB"| = |MN|$. By a similar way, we have equalities of other corresponding sides $|AC"| = |MP|$ and $|B"C"| = |NP|$.

Equalities of corresponding angles are easily introduced by using preservations of angles of rotation and translation. We have the following proof context

```
...
assert (distance AB" = distance MN).
rewrite <-(@rotation_isometrie M M N M B' (angCons MN⃗ AB⃗));
  auto with geo.
apply (@translation_isometrie M B' A B"):auto with geo.
...

H1 : directSimilar A B C M N P
H2 : distance A B" = distance M N
H3 : distance A C" = distance M P
H4 : distance B" C" = distance N P
H5 : angleCons MN⃗ AB"⃗ = angleCons MN⃗ AB⃗
H6 : angleCons MP⃗ AC"⃗ = angleCons MN⃗ AB⃗
H7 : angleCons NP⃗ B"C"⃗ = angleCons MN⃗ AB⃗
..
------------------------------------(1/1)
exists k:R, k <> 0 /\ distance B C = k*distance N P /\
distance A B = k*distance M N /\ distance A C = k*distance M P .
```

Unfolding the definition of direct similarity in H1, we have $\widehat{\overrightarrow{NP}\overrightarrow{NM}} = \widehat{\overrightarrow{BC}\overrightarrow{BA}}$ and $\widehat{\overrightarrow{PM}\overrightarrow{PN}} = \widehat{\overrightarrow{CA}\overrightarrow{CB}}$. By calculations of angle, we can prove that $\widehat{\overrightarrow{MN}\overrightarrow{AB}} = \widehat{\overrightarrow{MP}\overrightarrow{AC}} = \widehat{\overrightarrow{NP}\overrightarrow{BC}}$. Using this equation in H6 H7 give us

```
H1 : directSimilar A B C M N P
H2 : distance A B" = distance M N
H3 : distance A C" = distance M P
H4 : distance B" C" = distance N P
H5 : angleCons MN⃗ AB"⃗ = angleCons MN⃗ AB⃗
H6 : angleCons MP⃗ AC"⃗ = angleCons MN⃗ AB⃗
H7 : angleCons NP⃗ B"C"⃗ = angleCons MN⃗ AB⃗
..
------------------------------------(1/1)
exists k:R, k <> 0 /\ distance B C =k*distance N P /\
distance A B = k*distance M N /\ distance A C = k*distance M P .
```

We replace the left parts by the right parts of the equalities in H2, H3 and H4. We then apply the above consequence of Thales' theorem. We have to prove 3 subgoals

```
...
rewrite <-H2, <-H3, <-H4.
apply Thales_consequence.

H1 : directSimilar A B C M N P
H5 : angleCons MN⃗ AB"⃗ = angleCons MN⃗ AB⃗
H6 : angleCons MP⃗ AC"⃗ = angleCons MN⃗ AB⃗
H7 : angleCons NP⃗ B"C"⃗ = angleCons MN⃗ AB⃗
..
------------------------------------(1/3)
col A B B"
------------------------------------(2/3)
col A C C"
------------------------------------(3/3)
parallel (line B C) (line B"C")
```

We observe that collinearity of points A, B and B" is defined by collinearity of vectors $\exists k, \overrightarrow{AB"} = k\overrightarrow{AB}$); similarly collinearity of points A, C and C" is defined by collinearity of vectors $\exists k, \overrightarrow{AC"} = k\overrightarrow{AC}$) and parallelism of lines BC and B"C" is defined by $\exists k, \overrightarrow{BC} = k\overrightarrow{B"C"}$). As a result, all three sub-goals are solved by proving the following lemma for non-zero vectors:

$\widehat{\overrightarrow{u}\overrightarrow{v_1}} = \widehat{\overrightarrow{u}\overrightarrow{v_2}} \rightarrow \exists k, \overrightarrow{v_1} k \overrightarrow{v_2}$

The proof of this lemma is performed by unfolding the angles to records of scalar products and using equalities of corresponding elements of the records.

.....End of Technical Details.....

3.5.2 Orientation in Proving Theorems

We now give some applications to the proof of geometry theorems by proving 2 theorems in the list of famous theorems in mathematics [19]. We will show the indispensability of notion and the applications of its properties in these proofs.

3.5.2.1 Ptolemy's Theorem

Ptolemy's theorem describes a relation in Euclidean geometry between the four sides and two diagonals of a cyclic quadrilateral (a quadrilateral whose vertices lie on a common circle). Let a convex quadrilateral ABCD be inscribed in a circle, we have $|AD| \times |BC| + |AB| \times |CD| = |AC| \times |BD|$

Statement using orientation: Firstly we describe the statement of this theorem in Coq. One question is how to present a convex quadrilateral. Formally, a convex

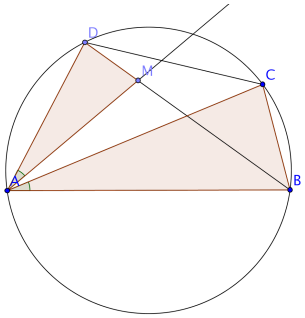


Figure 3.9: Demonstration of Ptolemy's theorem

polygon is a polygon which satisfies one of the following conditions : the entire segment connecting two points is contained in a polygon if two points are contained in this polygon, every interior angle is $\leq \pi$... We use an equivalent definition: a polygon is convex if and only if all ordered triplets points in the enumerated order have the same orientation. In our case, a convex quadrilateral ABCD is described by

Definition convexQuad (A B C D :Point):=
$\circlearrowleft\circlearrowleft ABCBCD \wedge \circlearrowleft\circlearrowleft ABCCDA \wedge \circlearrowleft\circlearrowleft ABCDAB$

Thus, the statement of Ptolemy's theorem is as follows:

Theorem Ptolemy: $\forall(A\ B\ C\ D : Point)$,
$convexQuad\ A\ B\ C\ D \rightarrow co-cyclic\ A\ B\ C\ D \rightarrow |AB| \times |CD| + |BC| \times |DA| = |AC| \times |BD|$.

Proof: The proof of this theorem is realized by locating a point M on BD such that $\widehat{\overrightarrow{AB}\overrightarrow{AC}} = \widehat{\overrightarrow{AM}\overrightarrow{AD}}$ (see Fig. 3.9) . We readily get that $\overrightarrow{AM}$ is between $\overrightarrow{AB}$ and $\overrightarrow{AD}$, in other words M is in interior of angle $\widehat{\overrightarrow{AB}\overrightarrow{AD}}$. So the four points A, B, D and M satisfy the configuration of property 8. It follows that the intersection point of AM and BD is inside segment BD, so we have $|BM| + |MD| = |BD|(1)$.

Consider $\triangle ABC$ and $\triangle AMD$, from hypothesis of locating M we have that $\widehat{\overrightarrow{AB}\overrightarrow{AC}}$ = $\widehat{\overrightarrow{AM}\overrightarrow{AD}}(2)$. Thanks to property 10, with two inscribed angles $\widehat{\overrightarrow{CA}\overrightarrow{CB}}$ and $\widehat{\overrightarrow{DA}\overrightarrow{DB}}$ which intercept the same arc $\overset{\frown}{AB}$ and they have the same orientation $\circlearrowleft\circlearrowleft CABDAB$ (proved from hypotheses), we get $\widehat{\overrightarrow{CA}\overrightarrow{CB}} = \widehat{\overrightarrow{DA}\overrightarrow{DB}}(3)$. By the fact that M is in segment [BD] we get $\widehat{\overrightarrow{DA}\overrightarrow{DB}}$=$\widehat{\overrightarrow{DA}\overrightarrow{DM}}(4)$, it follows that $\widehat{\overrightarrow{CA}\overrightarrow{CB}} = \widehat{\overrightarrow{DA}\overrightarrow{DM}}(5)$. Let's consider 2 triangles $\triangle ABC$ and $\triangle AMD$, (2) (5) show that they have 2 pair of congruent angles. So by definition of similar triangles, they are similar $\triangle ABC \sim \triangle AMD$. We get $\frac{|AB|}{|AM|} = \frac{|BC|}{|MD|} = \frac{|CA|}{|DA|}$, hence $|BC| \times |AD| = |MD| \times |AC|$ (6).

Similarly with $\triangle ACD$ and $\triangle ABM$, we have $\frac{|AC|}{|AB|} = \frac{|CD|}{|BM|} = \frac{|DA|}{|MA|}$ and $|AB| \times |CD| = |BM| \times |AC|$ (7).

By adding equations (6) (7) and applying (1) we have $|AB| \times |CD| + |BC| \times |AD| = |BM| \times |AC| + |MD| \times |AC| = |AC| \times |BD|$. □

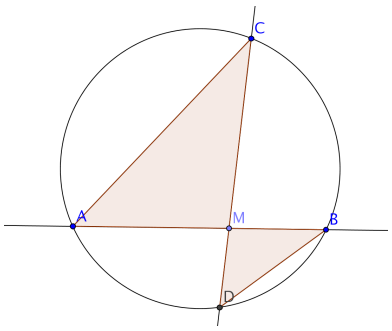


Figure 3.10: Intersecting Chords Segment Product property

The theorem is proved with the help of orientation by applying properties 10 and 8. Without property 8, we can not prove the existence of M inside BD, it follows that we do not have the equations (1) and (4). Without property 10, we do not have the equality of inscribed angles, so we do not have the equation (3). Then, the proof is not complete.

3.5.2.2 Product of Segments of Chords

Another geometry theorem we would like to present is the Product of Segments of Chords. In fact, it is one of three theorems in a series dealing with chords, secants and tangents in circles. Given four points A, B, C and D which lie on the same circle such that AB and CD intersect in M, the theorems state the equality $|MA| \times |MB| = |MC| \times |MD|$. If point M is interior to the circle (in other words, is interior to the segments AB and CD), this is called the Chords theorem. If it is exterior we have the Secants theorem. In a particular case of Secants theorem where two points of a secants coincide,we have the Tangents theorem. This is also the theorem about the power of a point with respect to a circle. However, we present here its traditional proof where order notion is used.

Chords theorem : If two chords intersect at an interior point of a circle, the product of the lengths of the segments of one chord equal the product of the segments of the other $|MA| \times |MB| = |MC| \times |MD|$.

Statement of the theorem: Before explaining how to prove this in Coq, we need to clarify the notion 'two chords intersect at a interior point of a circle'. A point in the interior of a circle can be presented by existence of two points on the circle, between which this point lies.

```
Definition insideCircle (M :Point) (c :Circle):=
∃(I J :Point),liesOnCircle I c ∧liesOnCircle J c ∧between I J M.
```

Chords theorem is described as follows:

```
Theorem Chords: ∀( A B C D M : Point)(c : Circle),co − cyclic A B C D → c =
```

$(circle3points\ A\ B\ C) \rightarrow insideCircleMc \rightarrow collinear\ A\ B\ M \rightarrow collinear\ C\ D\ M \rightarrow |MA| \times |MB| = |MC| \times |MD|$.

Proof: The traditional proof using similar triangles is easy. We use similarity of 2 triangles $\triangle MAC \sim \triangle MDB$. However, to obtain this similarity, we have to determine the relative position of M with respect to AB and CD. It means that we need to show that M is interior to the both segments AB and CD. The following property is a consequence of property 8 for the case of 4 co-cyclic points.

Property Orient 12. : Given two chords of a circle, if they intersect at a point which is in the interior of one chord, this point will be in the interior of the other.

Lemma intersection_Chords: $\forall(\ A\ B\ C\ D\ M\ Point), co - cyclic\ A\ B\ C\ D \rightarrow between\ A\ B\ M \rightarrow collinear\ C\ D\ M \rightarrow between\ C\ D\ M$.

Coming back to the proof, by hypothesis that M is in the interior of the circle, we can deduce the existence of points I J on the circle such that M is in segment IJ. By applying property 12 with two chords IJ and AB, we have that M is in segment AB. This property give us $\widehat{\overrightarrow{AM}\overrightarrow{AC}} = \widehat{\overrightarrow{AB}\overrightarrow{AC}} = \widehat{\overrightarrow{MB}\overrightarrow{MC}}$. Similarity with IJ and CD, we get that M is in segment CD, then we have $\widehat{\overrightarrow{DB}\overrightarrow{DM}} = \widehat{\overrightarrow{DB}\overrightarrow{DC}} = \widehat{\overrightarrow{MB}\overrightarrow{MC}}$ From the two equations above, we deduce $\widehat{\overrightarrow{AM}\overrightarrow{AC}} = \widehat{\overrightarrow{DB}\overrightarrow{DM}}$. By the same way, we have $\widehat{\overrightarrow{CA}\overrightarrow{CM}} = \widehat{\overrightarrow{BM}\overrightarrow{BD}}$. So $\triangle MAC \sim \triangle MDB$ and we have $\frac{|MA|}{|MC|} = \frac{|MD|}{|MB|}$ or $|MA| \times |MB| = |MC| \times |MD|$. □

3.6 Some discussions about full angles

Let'us now discuss about a remarkable method - the full angles method [12]. As mentioned, this is a coordinate free method relying on a geometric invariant called the full-angle to prove theorems. This can generate short, human-readable and diagram independent proofs.

The most interesting of this method lies in the crucial notion of the full-angle. The authors try to provide a notion about angles which is similar to oriented angles of vectors, in order to construct traditional proofs without involving order relations. In fact, full-angles are oriented angles of 2 lines (instead of two vectors as in definition of oriented angle of vectors), this notion is robust to solve problems concerning inscribed angle, similar triangles, etc.

For example, in Fig. 3.6, the oriented angle of lines DA and DB (denoted $\measuredangle\ DA\ DB$) is always equal $\measuredangle\ CA\ CB$ without knowledge about relative position of D.

We can define similarity of 2 triangles by equality of corresponding oriented angle of line without specifying "irect similar" or "indirect similar". Furthermore we can easily prove the problem of Product of Segments of Chords using this notion.

However, as a coordinate free method, this can not solve problems in which order relation is essential. Intuitively, we can not solve the problem of Ptolemy with this

method because it can not ensure that M lies inside BD (see the above proof of this theorem). In spite of this, formalizing this notion is still interesting for the advantages that it offers in treatment of inscribed angle and similar triangles.

3.7 Conclusion

In this chapter, we formalized plane orientation. Its relation with oriented angle was introduced, this allows us to remove ambiguities in the presentation of inscribed angles. Besides, the relations between orientation and order were approached. They form a toolbox to state and solve ordered geometry problems. This is tested through the proof of 2 famous theorems which had not been formalized in Coq before.

As mentioned, we aim at application in high school education, students somehow use our library to construct proofs. However, in high school geometry proofs, oriented problems in minor reasoning steps are omitted, as they lead to hard proving techniques and are pedagogically not suitable to students. So, some works remain to be done to improve this toolbox. Providing an automatic tactic to solve this kind of problems is a good research direction. It is considered as future work.

Chapter 4

Multiple Points of View about Geometry

4.1 Introduction

Synthetic or axiomatic geometry is the branch of geometry which makes use of axioms, as opposed to algebraic geometry. While synthetic proofs are traditional proofs composed by sequences of geometric reasoning steps that are based on geometric objects and their properties, algebraic proofs are constructed by algebraic calculations over polynomials of coordinates that describe geometric objects.

The mixture of synthetic and algebraic methods offers coordinate-free methods. Generated proofs of these methods are sequences of calculations over geometric invariants, such as the area of triangles or full angle of lines, that are intuitive. Therefore, proofs are partially traditional and human-readable.

Some development in Coq with the aim of formalizing geometry were previously performed. For axiomatic geometry, the systems of Hilbert and Tarski were respectively formalized by C. Dehlinger [16] and J.Narboux [40]. The development of J.Duprat [17] is to construct a library that contains an axiomatization of the ruler and compass for Euclidian geometry. Typically, following several chapters in the book *Meta-mathematische Methoden in der Geometrie* [47], J.Narboux mechanized proofs of over 150 lemmas in Tarski'geometry [41].

For algebraic geometry, there are efforts of J.Narboux and L.Pottier in formalizing coordinates-free and algebraic methods such as the area method [37] and the method of Gröbner bases [33][45] in Coq. These methods are implemented in the form of tactics, and allow us to prove many theorems.

However, there is not, prior to our work, any formal link between these systems. This motivates us to make our library to be a foundation to connect these systems. We integrate the systems into our library. With this integration, we can make sure that the

concepts that we have formalized are consistent with the concepts in the other systems. Moreover, it allows us to use automatic proofs in the other systems and to interleave them with interactive proofs in our system.

Related work: The idea of using a common formal development about geometry for different purpose was also proposed by J.Narboux in [41]. He would like to integrate the development of F.Guilhot and his development about the area method into Tarski'system. His direction has an advantage in using the axiomatic system of Tarski which is standard and has low level with respect to the axiomatic system of our library which comes from school's curriculum. However, this low level leads to difficulties in integrating. In my knowledge, his work has not been finished yet.

4.2 Verifying axiomatic systems of synthetic geometry

In this section, we consider Hilbert's and Tarski's systems. We do not intend to formalize proofs of theorems presented in these systems, we merely want to verify that their axioms are provable in our axiomatic system. We focus on axioms for plane geometry.

4.2.1 Verifying Hilbert's system

Hilbert proposed his axiomatic system for Euclidean geometry in [30]. His axioms state facts about primitive notions such as points, lines and planes and relations such as betweeness, congruence and lying on. He classified them into five groups: I.Incidence axioms, II.Order axioms, III.Congruence axioms, IV.Parallelism axiom and V.Continuity axioms.

The first step of the process of verifying his axioms is to make his primitive notions and relations available in our system. In fact, these notions and relations are the elementary ones that are covered in our library. As a result, we can work directly on the axioms. The following are primitive notions of Hilbert in our system:

- Point: primitive notion
- Line: defined by a record containing a root point and a non-null direction vector (rootPoint, $\overrightarrow{dirVect}$, $\overrightarrow{dirVect} \neq \overrightarrow{0}$). A line passing through two distinct points A and B is defined by (A, $\overrightarrow{AB}$).
- Point lying on a line: defined by collinearity of the direction vector of the line with the vector composed by the root point and this point $A \in (P, \overrightarrow{v}) := \exists k, \overrightarrow{AP} = k\overrightarrow{v}$
- Point lying between 2 distinct points: this is equivalent with the "between" notion mentioned on page 56 of Chapter 3. Essentially this notion can be expressed using vectors as follows: $C \in [AB] := A \neq B \wedge \exists k, 0 < k \wedge k < 1 \wedge \overrightarrow{AC} = k\overrightarrow{AB}$

- Congruence of 2 segments: defined by equality of distances of segments $|AB| = |CD|$ where distance of 2 points A and B is defined by scalar product as follows $|AB| := \sqrt{\overrightarrow{AB} \cdot \overrightarrow{AB}}$
- Congruence of 2 triangles: defined equality of their corresponding sides $\triangle ABC = \triangle A'B'C' := |AB| = |A'B'| \wedge |BC| = |B'C'| \wedge |CA| = |C'A''|$

Before going into each group, we note that Hilbert's axioms are always stated in the form of "for every x". This means that they are true in every case including degenerated cases where primitives in statement are identical (for example equality of points, lines in statements). However, in the development of his system, he usually made distinction of primitive objects which have the same type (such as point, line, etc). Therefore, some non-degenerated conditions are added in axiom statement in Coq. We prove the axioms under these conditions. However, proofs of the degenerate cases are easily performed. The continuity axioms require second-order logic and have not been treated in our system yet.

4.2.1.1 Axioms of Incidence

The axioms of this group for plane geometry are related to the notion of line and points lying on lines.

- Axiom I.1. For every two points A, B there exists a line a that contains each of the points A, B.
- Axiom I.2. For every two points A, B there exists no more than one line that contains each of the points A, B.
- Axiom I.3. There exist at least two points on a line. There exist at least three points that do not lie on a line.

The first two semantically state unique existence of a line that passes through two distinct points A and B. They are translated to two lemmas in Coq

Lemma AI_1 : $\forall(A\ B\ : Point), A \neq B \rightarrow \exists l : Line, A \in l \wedge B \in l.$
Lemma AI_2 : $\forall(A\ B : Point)(a\ b : Line), A \neq B \rightarrow A \in a \wedge B \in a \wedge A \in b \wedge B \in b \rightarrow a == b.$

Note that, as the line is expressed by a pair of point and direction vector, we use the semantic equality of lines (denoted ==) instead of syntactic one in the second lemma. The semantic equality, presenting coincidence of lines, is defined by collinearity (also called parallelism) of the direction vectors and vector composed by the root points as mentioned in Chapter 2

Proofs of these lemmas rely on a function named *line* in our library that takes A, B as arguments and return a line (A, $\overrightarrow{AB}$). We can easily prove that $A \in line(A\ B) \wedge B \in line(A\ B)$, hence the lemma AI_1 is solved by the existence of (line AB).

For the lemma AI_2, we prove that both of a and b are equal to *line ab*. In other words, we have to prove that $\forall a, A \in a \wedge B \in a \wedge A \neq B \rightarrow a == line(AB)$. Indeed, suppose that a = (X, $\overrightarrow{x}$), from the hypothesis $A \in a \wedge B \in a$, we have $\overrightarrow{XA} = k_1 \overrightarrow{x} \wedge \overrightarrow{XB} = k_2 \overrightarrow{x}$. By calculations over vector, we have $\overrightarrow{XA} = k_1 \overrightarrow{x} = \frac{k_1}{k_2 - k_1} \overrightarrow{AB}$. Thus, we have collinearity of $\overrightarrow{x}$, $\overrightarrow{XA}$, and $\overrightarrow{AB}$. This results in $(X, \overrightarrow{x}) == (A, \overrightarrow{AB})$, hence $a == line\ AB$. Similarly, we get $b == line\ AB$. So, the axiom AI_2 is proved by a == line(A B) == b. □

The statement of the lemma I.3 contains 2 part corresponding to the two following lemmas.

Lemma AI_3_1 : $\forall (a : Line), \exists\ A\ B, A \neq B \wedge A \in a \wedge B \in a.$
Lemma AI_3_2 : $\forall (A\ B : Point), A \neq B \rightarrow \exists C, \neg C \in line(A\ B).$

We prove the lemma AI_3_1 by assigning the root point of a to A, and constructing B such that $\overrightarrow{AB}$ is equal to the direction vector of a. The construction of such a point B is performed by *existence_multVectRepresentative* mentioned on page 1.

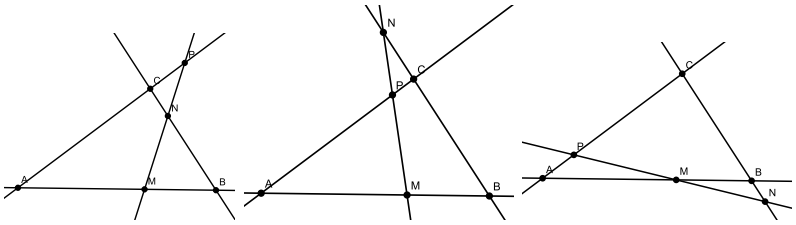
To prove the latter, we use the existence of three non-collinear points O O_1 and O_2 in our axiom system. As they are not collinear, so it is impossible that they lie on the line AB at the same time. So, we can chose one among them to assign to C. Using proofs by cases and contradiction, we easily get $\neg C \in line(A\ B)$.

4.2.1.2 Axioms of Order

- Axiom II.1. If a point B lies between a point A and a point C then the points A, B, C are three distinct points of a line, and B then also lies between C and A.
- Axiom II.2. For two points A and C, there always exists at least one point B on the line AC such that C lies between A and B.
- Axiom II.3. Of any three points on a line there exists no more than one that lies between the other two.
- Axiom II.4. Let A, B, C be three points that do not lie on a line and let a be a line in the plane ABC which does not meet any of the points A, B, C. If the line a passes through a point of the segment AB, it also passes through a point of the segment AC, or through a point of the segment BC (see Fig. 4.1).

The fact that betweenness $C \in [AB]$ (also denoted by $\overline{ACB}$ or *between ACB*) is expressed by $0 < k \wedge k < 1 \wedge \overrightarrow{AC} = k\overrightarrow{AB}$ allows us to readily prove the first three axioms.

The proof of the last axiom is the most difficult and interesting, in that the notion of orientation that we formalize in Chapter 3 plays a crucial role. Let's see the statement of this axiom in Coq



(a) N lies inside BC (b) C lies inside NB (c) B lies inside CN

Figure 4.1: 3 cases of axiom II.4

Lemma AII_4 :
$\forall(A\ B\ C : Point)(l : Line), \neg A \in l \rightarrow \neg B \in l \rightarrow \neg C \in l \rightarrow \neg colABC \rightarrow (\exists M, M \in l \wedge M \in [AB]) \rightarrow (\exists N, N \in l \wedge N \in [BC]) \vee (\exists P, P \in l \wedge P \in [AC])$.

.....Begin of Technical Details..... 7.
Proof of axiom II.4: Let's consider two lines BC and l that passes through M, N and P.

For the special case that the line l passing through M is parallel with BC, we prove that the intersection P of this line with line AC lies in segment AC. This is straightforward because in $\triangle ABC$ we have that $M \in [AB]$ and $MP \parallel BC$. So we have $\frac{\overrightarrow{AM}}{\overrightarrow{AB}} = \frac{\overrightarrow{AP}}{\overrightarrow{AC}}$.

We now consider the case that l intersects BC. Suppose that N is the intersection point. Once again, the method of proof by cases is used. We devise this problem into 3 cases illustrated in Fig. 4.1 that is corresponding with $\overline{CNB}$, $\overline{NCB}$ and $\overline{CBN}$. In the first case where N lies inside BC, we can finish our proof. The last two are proved using the same technique that relies on the orientation and its properties, so we present only the part of proof for the case where C lies between N and B ($\overline{NCB}$). This case corresponds to Fig. 4.1(b).

Before delving into the details of the proof, we remind the readers of some properties related to orientation mentioned in Chapter 3 that will be used in this proof.

- Property Orient6: $\neg col\ ABC \rightarrow \overrightarrow{ABD} \rightarrow\circlearrowleft\circlearrowleft\ ABCADC$
- Property Orient6' (a variant of property 6): $\neg col\ ABC \rightarrow \overrightarrow{ACD} \rightarrow\circlearrowleft\circlearrowleft\ ABCABD$
- Property Orient7: $\neg col\ ABC \rightarrow \overline{DAB} \rightarrow\circlearrowleft\circlearrowleft\ ABCACD$
- Property Orient7' (a variant of property 7): $\neg col\ ABC \rightarrow \overline{DAC} \rightarrow\circlearrowleft\circlearrowleft\ ABCADB$
- Property Orient8: $interior\ ABCD \rightarrow \exists E, \overline{BEC} \wedge \overrightarrow{ADE}$

The proof context with $\overline{NCB}$ is as follows

```
H : ¬ col A B C
H0 : liesOnLine M l
H13 : liesOnLine N l
H1 : between A M B
H15 : between N C B
______________________________________(1/2)
(exists P : Point, liesOnLine P l /\ between A P C) \/
(exists N : Point, liesOnLine N l /\ between B N C)
```

In this context, there is a second goal that is for the case $\overline{CBN}$.

Returning to our proof, we will prove that the intersection point P of AC and MN satisfies $liesOnLine\ E\ l \wedge \overline{APC}$. It seems that the configuration of N, A, C, M and P satisfies property *Orient8*. Using this property gives us $\overline{APC} \wedge \overrightarrow{NMP}$ so that our current goal $liesOnLine\ P\ l \wedge \overline{APC}$ can readily be proved.

However, these new hypotheses only are introduced after we prove $interior\ NACM$ that is the hypothesis of the property *Orient8* to ensures a correct application of this property. The commands in Coq are as follows

```
...
left.
destruct(@Orient8 N A C M) as [P [H16 H17].
2:exists P.
```

```
H : ¬ col A B C
H0 : liesOnLine M l
H13 : liesOnLine N l
H1 : between A M B
H15 : between N C B
...
______________________________________(1/3)
interior NACM
______________________________________(2/3)
liesOnLine P l /\ between A P C
```

Unfolding the definition of "interior" leads us to prove $\circlearrowleft\circlearrowleft\ NAMNAC \wedge \circlearrowleft\circlearrowleft\ NMCNAC$. Let's see in the hypotheses, we have $\overline{AMB}$, $\overline{NCB}$. So $\circlearrowleft\circlearrowleft\ NAMNAC$ and $\circlearrowleft\circlearrowleft\ NMCNAC$ are proved by sequence of application of the above properties and the transitivity of orientation.

The proof of $\circlearrowleft\circlearrowleft\ NAMNAC$ is as follows: First, we assert $\circlearrowleft\circlearrowleft\ NAMNAB$ by applying property *Orient6* with A, M, N, B and the hypothesis H1. Similarly, we then assert $\circlearrowleft\circlearrowleft\ NABNAC$ by applying property *Orient6'* with N, A, B, C and H15.

```
...
unfold interior; split.
assert (H18:sameOrientation N A M N A B) by
```

```
 apply (@Orient6 A M N B H1)
assert (H19:sameOrientation N A B N A C) by
 apply (@Orient6' N A B C H15)

H1 : between A M B
H15 : between N C B
H18 : sameOrientation N A M N A B
H19 : sameOrientation N A B N A C
...
_____________________________________(1/4)
sameOrientation N A M N A C
_____________________________________(1/4)
sameOrientation N M C N A C
```

From the two new hypotheses ↺↺ $NAMNAB$ and ↺↺ $NABNAC$, we deduce ↺↺ $NAMNAC$ by the transitivity of orientation. This is performed by tactic *auto* with a data base containing this transitivity.

```
...
eauto with geo.

H : ¬ col A B C
H0 : liesOnLine M l
H13 : liesOnLine N l
H1 : between A M B
H15 : between N C B
...
_____________________________________(1/3)
sameOrientation N M C N A C
_____________________________________(2/3)
liesOnLine P l /\ between A P C
```

The proof of ↺↺ $NMCNAC$ is performed by a similar manner. Once it is proved, the application of Property 8 is taken into account. The two hypotheses H16 and H17 are introduced as in the following context.

```
...
eauto with geo.

H : ¬ col A B C
H0 : liesOnLine M l
H13 : liesOnLine N l
H1 : between A M B
H15 : between N C B
H16 : between A P C
H17: sameSide N M P
```

```
...
______________________________________(1/2)
liesOnLine P l /\ between A P C
______________________________________(2/2)
(exists P : Point, liesOnLine P l /\ between A P C) \/
(exists N : Point, liesOnLine N l /\ between B N C)
```

The left part of the current goal is readily solved thanks to H0, H13, and H17 and the right part is exactly H16. So the proof for the case $\overline{NCB}$ is finished.

.....End of Technical Details.....

4.2.1.3 Axioms of Congruence

- Axiom III.1. If A, B are two points on a line a, and A' is a point on the same or on another line a' then it is always possible to find a point B' on a given side of the line a' such that AB and A'B' are congruent.
- Axiom III.2. If a segment A'B' and a segment A"B" are congruent to the same segment AB, then segments A'B' and A"B" are congruent to each other.
- Axiom III.3. On a line a, let AB and BC be two segments which, except for B, have no points in common. Furthermore, on the same or another line a', let A'B' and B'C' be two segments which, except for B', have no points in common. In that case if AB=A'B' and BC=B'C', then AC=A'C'.
- Axiom III.4. If $\angle ABC$ is an angle and if B'C' is a ray, then there is exactly one ray B'A' on each "side" of line B'C' such that $\angle A'B'C' = \angle ABC$.
- Axiom III.5. (Side-Angle-Side) If for two triangles ABC and A'B'C' the congruences AB=A'B', AC=A'C' and $\angle BAC = \angle B'A'C'$ are valid, then the congruence $\triangle ABC = \triangle A'B'C'$ is also satisfied.

Axiom III.1 can be verified by constructing B' on a' such that $\overrightarrow{A'B'} = \overrightarrow{AB}$. This construction also relies on the barycenter function over mass points as in the proof of lemma I.3 above. Axiom III.2 is easily achieved because we use the real number as measure of segment. In Axiom III.3, the hypothesis of having no points in common except for B of 2 segments AB and BC is translated to relation that B lies between A and C. Similarly, we have B' is between A' and C'. So, the axiom is stated as $\overline{ABC} \wedge \overline{A'B'C'} \wedge |AB| = |A'B'| \wedge |BC| = |B'C'| \rightarrow |AC| = |A'C'|$. The hypotheses allow us to prove $|AC| = |AB| + |BC|$ and $|A'C'| = |A'B'| + |B'C'|$, hence $AC = A'C'$

Axiom AIII.4 is illustrated in Fig. 4.2. Requirement of "each side" leads us to the use of same orientation for its formal representation. We have to prove existence of 2 points A'_1 and A'_2 such that $\circlearrowleft\circlearrowleft\ B'A'_1C'B'C'A'_2$ and $\angle A'B'_1C' = \angle A'B'_2C' = \angle ABC$.

However, orientation and oriented angle have similar meanings, equality of oriented angle can lead to the same orientation. Thus, such A'_1 and A'_2 on opposite sides of B'C' can be expressed by $\widehat{\overrightarrow{BA}\overrightarrow{BC}} = \widehat{\overrightarrow{B'A'_1}\overrightarrow{B'C'}} = \widehat{\overrightarrow{B'C'}\overrightarrow{B'A'_2}}$. In other words, we prove the existence of A'_1 and A'_2 satisfying $\widehat{\overrightarrow{B'A'_1}\overrightarrow{B'C'}} = \widehat{\overrightarrow{BA}\overrightarrow{BC}}$ and $\widehat{\overrightarrow{B'A'_1}\overrightarrow{B'C'}} = -\widehat{\overrightarrow{BA}\overrightarrow{BC}}$. As a result, this axiom is translated into the following lemmas

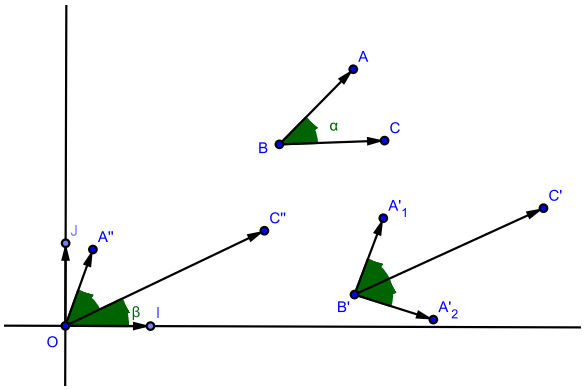


Figure 4.2: Figure of axiom III.4

```
Lemma AIII_4_case1: forall A B C B' C' :Point,
B<>A -> B <>C -> B' <> C' -> ¬ col A B C ->
exists A', ∠BAC = ∠B'A'C'
```

```
Lemma AIII_4_case2: forall A B C A' B' :Point,
B<>A -> B <>C -> B' <> C' -> ¬ col A B C ->
exists A', -∠BAC = ∠B'A'C'.
```

Instead of proving these two lemmas, we prove an auxiliary lemma that is their generalization as follows: given two distinct points B', C', for every angle α, there is A' such that $\widehat{\overrightarrow{B'A'}\overrightarrow{B'C'}} = \alpha$. It's not hard to find that by a translation of the points that moves B' into the root O of the coordinates system OIJ, this lemma is transferred into constructing A'' such that $\widehat{\overrightarrow{OA''}\overrightarrow{OC''}} = \alpha$, where C'' is image of C' by the translation (see Fig. 4.2).

Let's consider coordinates of A" in Fig 4.2. Suppose that $\widehat{\overrightarrow{Ox}\overrightarrow{OC''}} = \beta$, we have $(x_{A''}, y_{A''}) = (\cos\widehat{\overrightarrow{Ox}\overrightarrow{OA''}}, \sin\widehat{\overrightarrow{Ox}\overrightarrow{OA''}}) = (\cos(\widehat{\overrightarrow{Ox}\overrightarrow{OC''}} + \widehat{\overrightarrow{OC''}\overrightarrow{OA''}}), \sin(\widehat{\overrightarrow{Ox}\overrightarrow{OC''}} + \widehat{\overrightarrow{OC''}\overrightarrow{OA''}})) = (\cos(\alpha+\beta), \sin(\alpha+\beta)) = (\cos\alpha\cos\beta - \sin\alpha\sin\beta, \sin\alpha\cos\beta + \cos\alpha\sin\beta)$. Because α and β are determined, $x_{A''}$ and $y_{A''}$ are also determined. Therefore, A" is constructed as follows

$\overrightarrow{OA''} = (\cos\alpha\cos\beta - \sin\alpha\sin\beta)\overrightarrow{OI} + (\sin\alpha\cos\beta + \cos\alpha\sin\beta)\overrightarrow{OJ}$

Finally, A' of this auxiliary lemma is constructed by $\overrightarrow{B'A'} = \overrightarrow{OA''}$. Application of this lemma with $\alpha = \pm\widehat{\overrightarrow{BA}\overrightarrow{BC}}$ gives us A'_1 and A'_2.

The last axiom in this group is about triangle equality. From the hypotheses, we have to prove that corresponding sides and angles of these two triangle are equal. This is performed thanks to the metric relations in a triangle that are provided in our library. For this case, we use $|BC|^2 = |AB|^2 + |AC|^2 - 2|AB||AC| \cos \overrightarrow{AB}\overrightarrow{AC}$.

4.2.1.4 Axiom of Parallels

- Axiom IV.1. Let a be any line and A a point not on it. Then there is at most one line in the plane that contains a and A that passes through A and does not intersect a.

In our formalization, we defined lineP(A a) that passes through A and is parallel with a by a line that takes A as its root point direction vector of a as its direction vector. Uniqueness of this line is verified. This allows us to easily prove this axiom.

4.2.2 Verifying Tarski's system

The history of Tarski's axiom systems is presented by J.Narboux in [41]. The final version of the axiomatic system was used in his formalization. With the aim of making homogeneous geometry formalizations in Coq, we also want to verify this axiomatic system. The axioms are based on two predicates : betweenness and equidistance. Whereas equidistance describes equality of distance in the same manner as the congruence notion in Hilbert's system, there is a difference with betweenness. This notion in Tarski's system takes coincidence of points into account. In particular, we say that B lies between A and C if A, B and C are the same or if B lies inside segment A C with A and C included for the case A and C are different. Use β to denote this notion, we have $\beta ABC \leftrightarrow A = B = C \vee (A \neq C \wedge (B = A \vee B = C \vee \overline{ABC})$. Axioms in [41] are stated for plane geometry as follows

- 1.Identity $\beta ABA \rightarrow A = B$
- 2.Pseudo-transitivity $AB = CD \wedge AB = EF \rightarrow CD = EF$
- 3.Symmetry $AB = BA$
- 4.Identity $AB = CC \rightarrow A = B$
- 5.Pasch $\exists X, \beta APC \wedge \beta BQC \rightarrow \beta PXB \wedge \beta QXA$
- 6.Euclid $\exists XY, \beta ADT \wedge \beta BDC \wedge A \neq D \rightarrow \beta ABX \wedge \beta ACY \wedge \beta XTY$
- 7.Five segments $AB = A'B' \wedge BC = B'C' \wedge AD = A'D' \wedge BD = B'D' \wedge \beta ABC \wedge \beta A'B'C' \wedge A \neq B \rightarrow CD = C'D'$
- 8.Construction $\exists E, \beta ABE \wedge BE = CD$

- 9.Lower Dimension $\exists ABC, \neg\beta ABC \wedge \neg\beta BCA \wedge \neg\beta CAB$
- 10.Upper Dimension $AP = AQ \wedge BP = BQ \wedge CP = CQ \wedge P \neq Q \rightarrow \beta ABC \vee \beta BCA \vee \beta CAB$
- 11.Continuity $\forall XY, (\exists A, (\forall xy, x \in X \wedge y \in Y \rightarrow \beta Axy)) \rightarrow \exists B, (\forall xy, x \in X \wedge y \in Y \rightarrow \beta xBy)$

As the betweenness notion inherently contains degenerated cases, the statements of axioms have a concise look. Moreover, proofs of theorem from this system do not need to be restricted to non-degenerate cases, they are more uniform. However, this makes proofs of axioms long with many case treatments. In addition, the proofs were realized using proof techniques similar to the ones of Hilbert's axioms. So, rather than addressing in details, we give the main ideas in proving some axioms which are as follows: For axiom 5, we use properties concerning the same orientation; For axiom 6, we construct line passing through T and is parallel with BC. This line respectively intersects AB, AC at X and Y; for axiom 10, we prove that for any point X satisfying $XP = XQ$, then X lies in perpendicular bisector of PQ, etc.

4.3 Combining with the area method

Our library contains a large number of geometric notions and propositions. This allows us to prove many geometry theorems. Constructing a traditional proof consists in finding a sequence in logical steps to the conclusion. But in the context of education, a drawback of constructing fully traditional proofs is that, as mentioned in the introduction section, there are minor goals in interactive proving which are necessary to complete the formal proof but which lead to tedious steps and are not adapted to the level of abstraction at which we usually work with students.

On the other hand, coordinate-free automatic deduction methods can automatically solve many non-trivial geometry theorems. Generated proofs are constructed by transformation, calculation of geometric invariant, hence human-readable.

So, beyond motivations about homogeneousness of formalization systems, integration of coordinate-free automatic deduction methods with our interactive proving library offers many advantages in a educational view. This integration allows us to realize proofs by switching between interactive proof and automatic proof. Constructed proofs are always human-readable and acceptable by students. This improves the power and does not decrease pedagogical meaning of the library.

4.3.1 The area method

The coordinate-free method we want to present here is the area method of Chou, Gao and Zhang [11]. This method provides a set of geometric constructions to construct

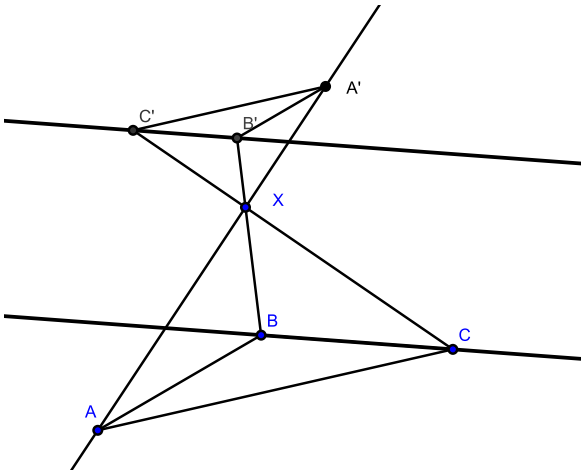


Figure 4.3: Parallelism of lines

the theorem and three geometric quantities: the signed area of a triangle, the signed distance and the Pythagoras difference to state the conjecture. Theorem statement is built by a sequences of constructions from free points. The conjecture is expressed by an equality between the geometric quantities or geometry relations. Geometry relations deeply is also expressed by equalities of the geometric quantities. The proof of the theorem is performed by eliminating all constructed points in reverse order of constructions, thanks to some provided elimination lemmas. This is repeated until there are only free points in the conjecture equality. The theorem is proved if the last equality is correct.

In fact, the elimination lemmas also have a form of an equality of the geometric quantities. Each lemma corresponds to one construction, where constructed points of the construction only appear in the left of this equality. So, applying this lemma allows us to remove a constructed point from the conjecture. These elimination lemmas are proved from an axiomatic system which are given in Table 4.1

We do not detail how this method works because this method was formalized in Coq by J.Narboux [41], and implemented it in the form of a tactic. It means that if we have a theorem statement with a set of constructions in hypotheses and a conjecture which satisfy requirement of the method, we can use this method to solve the theorem by a single command with this tactic. The following example shows the use of this method in Coq.

Example 2. :This example is illustrated in Fig 4.3.Given $\triangle ABC$ and $\triangle A'B'C'$. 3 lines AA', BB', and CC' are concurrent at X. If $AB \parallel A'B'$ and $AC \parallel A'C'$ then we have $BC \parallel B'C'$.

In the formalization of J.Narboux, this theorem is stated and solved with the area method as follows

```
Theorem Parallelisme :
```

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' 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 $P = P'$ (uniqueness)
9. If $PQ \parallel CD$ and $\frac{\overline{\overline{PQ}}}{\overline{\overline{CD}}} = 1$ then $DQ \parallel PC$ (parallelogram)
10. If $\mathcal{S}_{PAC} \neq 0$ and $\mathcal{S}_{ABC} = 0$ then $\frac{\overline{\overline{AB}}}{\overline{\overline{AC}}} = \frac{\mathcal{S}_{PAB}}{\mathcal{S}_{PAC}}$ (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 $4\mathcal{S}_{ABC}^2 = \overline{AF}^2\overline{BC}^2$ (area of a triangle)

Table 4.1: The axiom system for the area method

```
forall A B C X A' B' C' : Point, A' <>C' -> A' <> B' ->
on_line A' X A ->
on_inter_line_parallel B' A' X B A B ->
on_inter_line_parallel C' A' X C A C ->
parallel B C B' C'.
Proof.
area_method.
Qed.
```

Three constructions are used in statement with the order. A' lying on XA is constructed by *on_line A' X A*. B' satisfying $B' \in XB \wedge B'A' \parallel BA$ is constructed by *on_inter_line_parallel B' A' X B A B*. Finally, C' satisfying $C' \in XC \wedge C'A' \parallel CA$ is constructed by *on_inter_line_parallel C' A' X C A C*. The conjecture $BC \parallel B'C'$ is

expressed by *parallel BC B'C'*.

Constructions and relations of the area method are built from its primitive notions. They are not provided and used when stating theorems in our library. Therefore, to integrate this method, apart from proving its axiomatic system to ensure its correctness in our system, we have to convert statements from our system to the area method system to ensure usability of this method in alternative mode with interactive proving of our library.

4.3.2 Correctness of the area method

As the method has its own axiomatic system, correctness of the method is guaranteed by verifying its axioms. However, before going to verify the axiomatic system, we need to make a mapping of all notions that are used in these axioms into our library. Although this method relies on three geometric invariant including signed area, signed distance and Pythagoras difference, only signed area and signed distance are primitive notions. Pythagoras difference and other notions used in the axioms such as parallelism, perpendicularity, collinearity are defined from these two primitive notions.

Let's consider the notion of signed distance. This is a mixed notion from Euclidean distance and vector. It has magnitude as Euclidean distance and direction as vector. The difference lies in arithmetic operators. Addition in Chasles'axiom in Table 4.1 is an example : for every 3 collinear points A, B and C, $\overline{AC} = \overline{AB} + \overline{BC}$. This operator is similar to the addition operator of vector ($\forall\ ABC, \overrightarrow{AC} = \overrightarrow{AB} + \overrightarrow{BC}$) but this is only applied for segments which are collinear. It is similar to Euclidean distance but it takes into account the direction of segments. In particular, if $\overrightarrow{AB}$ and $\overrightarrow{BC}$ have the same direction then we have $|AC| = |AB| + |BC|$, if they have opposite directions we have $|AC| = ||AB| - |BC||$. This leads us to define signed distance by using Euclidean distance with sign constructed by the direction of vectors.

In Chapter 2, we showed the construction of a Cartesian coordinates system with O, I and J and how these points are used to define orientation for Euclidean plane. Here, they also allow us to define the sign of signed distance. Let's see a $\overrightarrow{AB}$ in the system OIJ that is represented in Fig 4.4. We move $\overrightarrow{AB}$ to the root O such that $A = O$. Intuitively, we define $\overline{AB} = |AB|$ if the image of B lies in the half plane on the right of the axis Oy (the point P in the figure), and $\overline{AB} = -|AB|$ if the image of B lies in the half plane on the left of the axis Oy (the point Q in the figure). The formal definition is as follows

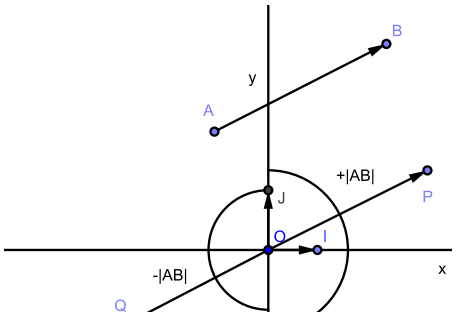


Figure 4.4: Definition of signed distance

$$\begin{cases} \text{if } \overrightarrow{AB} \cdot \overrightarrow{OI} > 0 & \overline{AB} = |AB| \\ \text{if } \overrightarrow{AB} \cdot \overrightarrow{OI} < 0 & \overline{AB} = -|AB| \\ \text{if } \overrightarrow{AB} \cdot \overrightarrow{OI} = 0 & \begin{cases} \text{if } \overrightarrow{AB} \cdot \overrightarrow{OJ} > 0 & \overline{AB} = |AB| \\ \text{if } \overrightarrow{AB} \cdot \overrightarrow{OJ} < 0 & \overline{AB} = -|AB| \\ \text{if } \overrightarrow{AB} \cdot \overrightarrow{OJ} = 0 & \overline{AB} = 0 \end{cases} \end{cases}$$

This definition expresses that its signed distance gets positive value if $\overrightarrow{AB}$ and $\overrightarrow{OI}$ have the same direction with respect to the axis Oy, and negative value if they have opposite directions. In the case they are orthogonal, we involve $\overrightarrow{OJ}$ in the definition in a similar way.

Intuitively, this definition sounds good because we easily find that $\overline{AB}$ and $\overline{BA}$ have different signs for every A and B. We move $\overrightarrow{AB}$ and $\overrightarrow{BA}$ to the root and get corresponding vectors $\overrightarrow{OP}$ and $\overrightarrow{OQ}$. The points P and Q are symmetric with respect to O, so if $\overline{OP} = |AB|$ then $\overline{OQ} = -|AB|$ and inversely. So we have $\overline{AB} = \overline{OP} = -\overline{OQ} = -\overline{BA}$. As a generalization of this, we can prove that $\forall ABC, \overrightarrow{AC} = k\overrightarrow{AB} \leftrightarrow (col\ ABC \wedge \overline{AC} = k\overline{AB})$. The proof is realized by dividing relative position of $\overrightarrow{AB}$ into cases cited in the signed distance definition, then using the value of k to determine relative position of $\overrightarrow{AC}$.

The remaining primitive notion is signed area. As in mathematic, the signed area of $\triangle ABC$ is defined by $\mathcal{S}_{ABC} = \frac{1}{2} \times |AB| \times |AC| \times \sin \overrightarrow{AB}\overrightarrow{AC}$. In the precedent chapter, the function sine is calculated by $\sin \overrightarrow{AB}\overrightarrow{AC} = \frac{\overrightarrow{AB}^{\perp} \cdot \overrightarrow{AC}}{|\overrightarrow{AB}^{\perp}| \times |\overrightarrow{AC}|}$. Therefore we have $\mathcal{S}_{ABC} = \frac{1}{2}\overrightarrow{AB}^{\perp} \cdot \overrightarrow{AC}$

Finally, we find equivalences for non-primitive notions including parallelism, perpendicularity and collinearity. Because these notions are also provided in our system, so this finding is straightforward. Using characters with small word "area" $\|_{area}$, $\perp_{area}$ and col_{area} to respectively denote these notions, we have the equivalence table 4.2

In the are method	In our system	Equivalence
$\overline{AB}$: primitive	$\overline{AB}$: is defined as above	—
$\mathcal{S}_{ABC}$: primitive	$\mathcal{S}_{ABC} := \frac{1}{2}\overrightarrow{AB}^{\perp} \cdot \overrightarrow{AC}$	—
$col_{area}\ ABC \leftrightarrow$ $\mathcal{S}_{ABC} = 0$	$col\ ABC \leftrightarrow$ $A = B \vee \exists k, \overrightarrow{AC} = k\overrightarrow{AB}$	$col_{area}\ ABC \leftrightarrow$ $col\ ABC$
$AB \parallel_{area} CD \leftrightarrow$ $\mathcal{S}_{ACB} + \mathcal{S}_{ABD} = 0$	$AB \parallel CD \leftrightarrow (A \neq B \wedge$ $C \neq D \wedge \exists k, \overrightarrow{AB} = k\overrightarrow{AC})$	$AB \parallel_{area} CD \leftrightarrow A = B \vee C = D$ $\vee (A \neq B \wedge C \neq D \wedge AB \parallel CD)$

Table 4.2: The Equivalence of notions

This table allows us to translated the axioms in Table 4.1 into our system. With this translation, we can perform proofs in our system by using all existening proved properties. We present here the proofs of some axioms

Axiom 4 If $\mathcal{S}_{ABC} = 0$ then $\overline{AB} + \overline{BC} = \overline{AC}$ (Chasles' axiom)

Proof: From equivalences in Table 4.2, we have $\mathcal{S}_{ABC} = 0 \rightarrow A = B \vee \exists k, \overrightarrow{AC} = k\overrightarrow{AB}$. For the case A=B, by replacing all occurrence of B by A in the proof obligations, we have $\overline{AB} + \overline{BC} = \overline{AC} \leftrightarrow \overline{AA} + \overline{AC} = \overline{AC} \leftrightarrow \overline{AC} = \overline{AC}$. □

Axiom 6 $\mathcal{S}_{ABC} = \mathcal{S}_{DBC} + \mathcal{S}_{ADC} + \mathcal{S}_{ABD}$ (dimension; all points are in the same plane)

Proof: By the definition of signed area, we have $\mathcal{S}_{ABC} = \mathcal{S}_{DBC} + \mathcal{S}_{ADC} + \mathcal{S}_{ABD} \leftrightarrow$
$\frac{1}{2}\overrightarrow{AB}^{\perp} \cdot \overrightarrow{AC} = \frac{1}{2}\overrightarrow{AB}^{\perp} \cdot \overrightarrow{AC} + \frac{1}{2}\overrightarrow{AB}^{\perp} \cdot \overrightarrow{AC} + \frac{1}{2}\overrightarrow{AB}^{\perp} \cdot \overrightarrow{AC} \leftrightarrow$
$\overrightarrow{AB}^{\perp} \cdot \overrightarrow{AC} = \overrightarrow{AB}^{\perp} \cdot \overrightarrow{AC} + \overrightarrow{AB}^{\perp} \cdot \overrightarrow{AC} + \overrightarrow{AB}^{\perp} \cdot \overrightarrow{AC}$

.

The last equality is already proved in the proof of property 4(on page 51). □

Axiom 10 If $\mathcal{S}_{PAC} \neq 0$ and $\mathcal{S}_{ABC} = 0$ then $\frac{\overline{AB}}{\overline{AC}} = \frac{\mathcal{S}_{PAB}}{\mathcal{S}_{PAC}}$

Proof: In the proof of this lemma, we suppose that axioms 2 and 3 are already proved. By applying these, we have $\mathcal{S}_{ACB} = \mathcal{S}_{BAC} = -\mathcal{S}_{ABC} = 0$. From equivalences in Table 4.2, we have $\mathcal{S}_{ACB} = 0 \rightarrow A = C \vee \exists k, \overrightarrow{AB} = k\overrightarrow{AC}$. For the case A=C, we have $\overrightarrow{AC} = \overrightarrow{0}$, hence $\mathcal{S}_{PAC} = \mathcal{S}_{ACP} = \frac{1}{2}\overrightarrow{AC}^{\perp} \cdot \overrightarrow{AP} = 0$. We have $\mathcal{S}_{PAC} \neq 0$ and $\mathcal{S}_{PAC} = 0$ at the same time, so the proof for this case is finished.

For the remaining case where $A \neq C \wedge \overrightarrow{AB} = k\overrightarrow{AC}$, by unfolding the definition of area, we have

$$\frac{\mathcal{S}_{PAB}}{\mathcal{S}_{PAC}} = \frac{\mathcal{S}_{ABP}}{\mathcal{S}_{ACP}} = \frac{\frac{1}{2}\overrightarrow{AB}^{\perp} \cdot \overrightarrow{AP}}{\frac{1}{2}\overrightarrow{AC}^{\perp} \cdot \overrightarrow{AP}} = \frac{\vec{AB}^{\perp} \cdot \overrightarrow{AP}}{\overrightarrow{AC}^{\perp} \cdot \overrightarrow{AP}} = \frac{k\overrightarrow{AC}^{\perp} \cdot \overrightarrow{AP}}{\overrightarrow{AC}^{\perp} \cdot \overrightarrow{AP}} = k$$

.

On the other hand, from a property of signed distance mentioned earlier, we have $\overrightarrow{AB} = k\overrightarrow{AC} \rightarrow \overline{AB} = k\overline{AC}$. Because $A \neq C \rightarrow \overline{AC} \neq 0$, we get $\frac{\overline{AB}}{\overline{AC}} = k$. From this transformation, we obtain $\frac{\mathcal{S}_{PAB}}{\mathcal{S}_{PAC}} = k = \frac{\overline{AB}}{\overline{AC}}$. □.

In a similar way, we prove all axioms of the area method. The formalization of this method is well integrated in our system. We can now go to the second step of our integration process.

4.3.3 Usability of the area method

The objective of this section is to make the area method on alternative mode to interactive proving with our library. In particular, in a context of interactive proof with hypotheses and a proof obligation, the area method can be invoked to solve it.

In fact, to be able to use the area method, as mentioned, there are requirements for theorem hypotheses and conjectures. This method has its own specific geometric constructions which are regarded as basic constructions. Hypotheses have to be stated as a sequence of these constructions. Similarly, a conjecture need to be expressed by certain provided geometric relation or equality of geometric invariant.

As a result, in this step, we have to find out construction sequence from the hypotheses in the proof context, and express the proof obligation in a form acceptable by the area method. The last point is similar to the first step of integration mentioned in the previous section, so we omit its detail and only focus on the process of finding constructions.

In our library as well as in mathematics, a geometry theorem can be stated either in constructive form or in predicate form, even in a mix of both. For example, the foot H of altitude AH in $\triangle ABC$ is the intersection point of BC with line passing through A and being perpendicular with BC. Simultaneously, it can be expressed by $col\ BCH \wedge AH \perp BC$.

Finding a construction sequence from hypotheses in predicate form is really hard. It's because there are many construction sequences that lead the same figure. For our example, we can consider it not only as construction of H from A, B and C, but also as construction of A from H, B and C as follows: we take a point H in BC; we then construct line passing through H that is perpendicular with BC; finally, we take a point A on the newly constructed line. This leads to an ambiguity in construction order of geometry objects that is forbidden in the area method.

In this section, we limit the scope of our work to a simpler case where theorem is stated in the constructive form. In this case, geometry objects are created by constructions provided by our library and the construction order is taken into account. Suppose that these constructions are always preserved in the proof context during interactive proofs. Thus, our work focuses on finding a sequence of constructions of the area method that represents the same theorem with the set of constructions of the statement.

To illustrate this process, we continue to use the example on page 83. The statement of this theorem in our system is as follows

```
1 subgoal
H1 : A' <> B'
..
H5 : liesOnLine A' (line A X)
H6 : b' == lineP A' (line A B)
H7 : c' == lineP A' (line A C)
H8 : B' = intersectionPoint b' (line B X)
H9 : C' = intersectionPoint c' (line C X)
..
______________________________________(1/1)
parallel_Line (line B C) (line B' C')
```

The constructions in this statement are interpreted as follows: we take a point A' on line AX (the hypothesis H5), we construct line b' passing through A' and parallel with AB (the hypothesis H6), we take B' as the intersection point of b' and BX, etc.

Our goal here is to convert this sequence of constructions to a sequence of construction for the area method that states the same theorem. The theorem is stated in the area method as follows:

```
Theorem Parallelisme :
 forall A B C X A' B' C' : Point, A' <>C' -> A' <> B' ->
 on_line A' X A ->
 on_inter_line_parallel B' A' X B A B ->
 on_inter_line_parallel C' A' X C A C ->
 parallel B C B' C'.
Proof.
area_method.
Qed.
```

We observe that the statement in constructive form is very close to the one in the area method. A main reason is that constructions of both systems have order, and they are closely related with elementary constructions by ruler and compass. As a result, converting constructions is performed by passing through these constructions. We divide our process into 3 following phases:

In the first phase, we normalize constructions in proof context. First, the constructions are recursively unfolded until there are only elementary constructions by ruler and compass in their representation. The elementary construction are arbitrary point, point on a line, line with 2 points, perpendicular line, parallel line, and intersection of lines, etc. This gives us sequences of the elementary constructions. We then perform some simplification of these sequences. We note that circles are barely treated in the area method [32], so they are not treated at this moment.

In our example, by using H6: $b' == lineP\ A'\ (line\ AB)$, this step replaces the hypothesis

H8: $B' = intersectionPoint\ b'\ (line\ BX)$

with

H8: $B' = intersectionPoint(lineP\ A'\ (line\ AB))(line\ BX)$

In the second phase, we translate the sequences that we get from the first phase into constructions of the area method. Since the constructions of the area method are close to the elementary constructions, for each construction, we can readily find out sequences of elementary constructions that express this constructions. Once one of sequences appears in the hypotheses after the first step, we try to convert it to this construction. Some constructions and their equivalent sequence are illustrated in Table 4.3.

Construction of the area method	Equivalent consequence of elementary constructions
on_line A B C	A ∈ BC
inter_ll I A B C D	I = AB ⋂ CD
on_parallel A' A B C	A' ∈ (lineP A BC)
on_inter_line_parallel Y R' U V P Q	Y = (lineP R' PQ) ⋂ UV
on_inter_parallel_parallel Y R' U V W P Q	Y = (lineP R' UV) ⋂ (lineP W PQ)
is_midpoint I A B	I = midpoint A B
on_perp A B C	A ∈ (lineT B BC)

Table 4.3: The equivalence of constructions

In our example, this phase converts the hypothesis

H8:$B' = intersectionPoint(lineP\ A'\ (line\ AB))(line\ BX)$

to the construction

H8: $on\_inter\_line\_parallel\ B'\ A'\ B\ X\ A\ B$.

Both of them express that we draw a line passing through A' and parallel with AB, this line intersects BX at B'.

In the last phase, we convert proof obligation in the form that is accepted by the area method.

.....Begin of Technical Details..... 8.

We now detail how we implement these phases in Coq.

Normalizing constructions The first phase is to normalize the constructions provided by our library. We try to have the simplest representations of construction.

We now unfold and replace compound constructions with corresponding sequences of the elementary constructions. For example, the perpendicular bisector of AB ($l == perpendiculer\_bisector\ AB$) is replaced by the line that passes through the middle point of AB and is perpendicular with AB ($l == lineT(midPoint\ AB)(line\ AB)$). As a result, we have only elementary constructions in the proof context. Our example does

not include compound constructions, hence this step does not have any effect in this case. The context proof remains unchanged.

Because constructions in the area method take only points as its arguments, we have to replace every occurrence of a line with its definition. As mentioned in Chapter 2, this replacement is also realized by the tactic *rewrite*. The following tactic allows us to automatically do this .

```
Ltac unfold_lines_to_points :=
repeat
match goal with
 |H : ?a == line _ _ |- _ =>
       rewrite H in *;revert H
 |H : ?a == lineT _ _ |- _ =>
       rewrite H in *;revert H
 |H : ?a == lineP _ _ |- _ =>
       rewrite H in *;revert H
end;
intros.
```

This tactic seeks all variant of definition of any line a in the hypotheses (the term a ==_). If it finds a hypothesis containing the definition (H: a ==_)), replacement with all occurrence of a in the context (*rewrite* H in *) is performed. This process of finding and replacing is repeated. To avoid infinite loops, after each step, we takes off treated hypothesis and put it in the goal (*revert* H). When the repeat loops finishes, we use *intros* to restore the hypotheses from the goal.

Finally, these sequences are reduced in a concise form without loosing their semantics. For example, a line passing through A that is parallel with perpendicular bisector of BC is simplified to a line passing through A that is perpendicular to line BC as follows: $l == linePA(perpendiculer\_bisector\ BC) \rightarrow l == linePA(lineT(midPoint\ BC)(line\ BC)) \rightarrow l == lineTA(line\ BC)$. Some lemmas need to be proved to ensure this simplification.

```
Lemma simplify_lineP_lineT :
forall A B C D :Point,
A <> B->
lineP D (lineT C (line A B)) == lineT D (line A B).
Lemma simplify_lineT_lineP :
forall A B C D :Point,
A <> B->
lineT D (lineP C (line A B)) == lineT D (line A B).
...
```

And a tactic is written to automatically do this simplification.

```
Ltac simplify_lineT_lineP :=
```

```
repeat
match goal with
|H:context [lineP ?D (lineT ?C (line ?A ?B))] |-_=>
      let H1:=fresh in
      cut (A<>B); [intros H1 | ];
      rewrite (@simplify_lineP_lineT A B C D H1) in *;
      clear H1
|H:context [lineT ?D (lineP ?C (line ?A ?B))] |-_=>
      let H1:=fresh in
      cut (A<>B); [intros H1 | ];
      rewrite (@simplify_lineT_lineP A B C D H1) in *;
      clear H1
..
end.
```

This tactic repeatedly finds hypotheses containing the construction *lineP D (lineT C (line A B))*, it then replace this construction by *lineT D (line A B)*.

So, our tactic in this part is a combination of the tactics above

```
Ltac normalizeStatement :=
...;
try unfold_lines_to_points;
try simplify_lineT_lineP.
```

After this phase, we have only combination of elementary constructions in a concise form. The proof context after applying this step is as follows:

```
Proof
normalizeStatement.

1 subgoal
H1 : A' <> B'
..
H5 : liesOnLine A' (line A X)
H8 : B' = intersectionPoint (lineP A' (line A B)) (line B X)
H9 : C' = intersectionPoint (lineP A' (line A C)) (line C X)
..
______________________________________(1/1)
parallel_Line (line B C) (line B' C')
```

Converting constructions In the second phase, we try to extract constructions of the area method from the sequence of constructions we get in the first step. Let's consider the constructions of the area method. In fact, each construction aims to create a geometric object with a precise semantics. For this, we know how to construct this object with elementary constructions.

For example, *on_inter_line_parallel* $Y\ R'\ U\ V\ P\ Q$ is the construction of Y such

that Y is the intersection point of line UV with the line passing through R' and parallel with line PQ. With the elementary construction, we can construct Y as follows: we first construct the lines PQ and UV; we then construct the line parallel to PQ that passes through R'; finally we take Y as intersection of the parallel line with line UV. Therefore, Y is defined by $Y = intersectionPoint\ (lineP\ R'\ PQ)\ UV$

This equivalence is assured by proving the following lemma. We need to find out all possible consequences for each construction of the area method and prove them.

```
Lemma constr_on_inter_line_parallel:
forall (Y R U V P Q : Point),
P <> Q ->
U <> V->
¬liesOnLine R (line P Q) ->
¬parallel_Line (line P Q)(line U V) ->
Y = intersectionPoint ( lineP R (line P Q)) (line U V) ->
on_inter_line_parallel Y R U V P Q .
```

For each construction of the area method, we know sequences of the elementary constructions that can construct the same object (see Tab. 4.3), we look for occurrences of these sequences in the proof context. Once we find them, we try to convert them into the equivalent construction of the area method.

```
Ltac convert_to_AMConstructions_6 :=
match goal with
| |- ?G =>
repeat
 match goal with
 |- _ -> G =>
  match goal with
  | H:?Y = intersectionPoint ( lineP ?R (line ?P ?Q)) (line ?U ?V)
   |- _ =>
   let H1 :=fresh in
   cut ( on_inter_line_parallel Y R U V P Q);
   [intros H1 |apply (@constr_on_inter_line_parallel Y R U V P Q);
      auto with geo ]
   ;revert H
  end
 | |- G =>
  match goal with
  | H:?Y = intersectionPoint ( lineP ?R (line ?P ?Q)) (line ?U ?V)
   |- _ =>
  let H1 :=fresh in
  cut ( on_inter_line_parallel Y R U V P Q);
  [intros H1 |apply (@constr_on_inter_line_parallel Y R U V P Q);
     auto with geo ]
```

```
  ;revert H
  end
 | _ =>  idtac "avoid"
  end
end;
intros.
```

A short description of this tactic is as follows:

- We use the structure *match goal with ... H:?Y = intersectionPoint..* to look for hypotheses of the form _ = *interesectionPoint (lineP _ _) _*.
- If we find such a hypothesis, we introduce its corresponding construction by using the tactic *cut (on_inter_line_parallel..)*.
- To prove equivalence of introduced construction with the former sequence, we apply its corresponding lemma *apply (@constr_on_inter_line_parallel ...)*
- We add some techniques to avoid loops

After this phase, constructions of the area method are introduced. However, some sub-goals are generalized. They are conditions necessary for the equivalence of constructions in the lemma *constr_on_inter_line_parallel*. The proof context is as follows:

```
Proof
normalizeStatement.
convert_to_AMConstructions.
1 subgoal
H1 : A' <> B'
..
H5 : liesOnLine A' (line A X)
H8 : B' = intersectionPoint (lineP A' (line A B)) (line B X)
H9 : C' = intersectionPoint (lineP A' (line A C)) (line C X)
H10 : on_inter_lineparallel B' A' B X A B
H12 : on_inter_lineparallel B' A' B X A B

..
______________________________________(1/7)
parallel_Line (line B C) (line B' C')
______________________________________(2/7)
~liesOnLine A' (line A C)
______________________________________(3/7)
~parallel_Line (line A C) (line C X)
```

Converting proof obligation This phase is to convert proof obligation into the form accepted by the area method. This phase is readily performed.

.....End of Technical Details.....

4.4 Combining with the method of Gröbner bases

In this method, geometry problems are stated in an algebraic form where points are represented by their coordinates and geometric predicates by polynomials. This method is performed by calculations over polynomials. The proof obligation will be constructed by a combination of polynomials in the hypotheses. Similar to the area method, this is formalized and provided under the form of a tactic *nsatz*. For more detail of the formalization of this method, we refer readers to [25].

In comparing with the area method, integration of this method is less complex. It is because we only need to translate our statements into the algebraic form. In particular, each hypothesis in our proof context has to be translated. We recall that there are 2 types for hypotheses: predicate form and constructive form. For example, the hypothesis that I *is the middle point of* AB can be stated as $|IA| = |IB| \wedge col\ A\ B\ I$ and $I = midPoint\ A\ B$.

Since we can get equivalent properties from geometry constructions, we only need consider the first type. In fact, geometric predicates can be represented by polynomials based on algebraic relations of vectors and scalar products, so geometric problems stated in the constructive form or the predicate form are easily turned into the algebraic form. Some translations are as follows

```
Lemma trans_col:
forall(A B C:Point), col A B C ->
(X A - X B)*(Y C - Y B)-(Y A - Y B)*(X C - X B)=0.
Lemma trans_parallel:
forall (A B C D:Point), parallel_Line (line A B)(line C D) ->
((X B)-(X A))*((Y D)-(Y C))=((Y B)-(Y A))*((X D)-(X C)).
Lemma trans_liesOn:
forall(A B C:Point), liesOnLine A (line B C) ->
(X B - X A)*(Y C - Y B)-(Y B - Y A)*(X C - X B)=0.
Lemma trans_perpendicular:
forall (A B C D:Point), perpendicular (line A B)(line C D) ->
  ((X B)-(X A))*((X D)-(X C))+((Y B)-(Y A))*((Y D)-(Y C))=0.
Lemma trans_samedistance:= forall (A B C D :Point)
distance A B = distance C D ->
(XB - XA)^2 + (YB - YA)^2 = (XD - XC)^2 + (YD - YC)^2.
```

Where X, Y are functions of type *Point -> Real*, that return the coordinates of the input point and defined in our Cartesian coordinate system OIJ system by
$X\ A := \overrightarrow{OA}\overrightarrow{OI}$
$Y\ A := \overrightarrow{OA}\overrightarrow{OJ}$
In fact, our formalization of the system OIJ makes easy calculations of point coordinates, vectors and scalar products. We saw a preview of this capability through technical detail 4 on page 55. Now we consider how to prove the above lemmas.

The following are some properties concerning point coordinates and vector that are easily verified and used in our proofs:
$\forall AB : Point, (XB - XA) = \overrightarrow{AB}\overrightarrow{OI} \wedge (YB - YA)\overrightarrow{AB}\overrightarrow{OJ}$
$\forall AB : Point, \overrightarrow{AB} = (XB - XA)\overrightarrow{OI} + (YB - YA)\overrightarrow{OJ}$
$\forall (ABCD : Point)(k : R), \overrightarrow{AB} = k\overrightarrow{CD} \rightarrow$
$(XB - XA) = k(XD - XC) \wedge (YB - YA) = k(YD - YC)$
$\forall (ABCD : Point)(k : R), \overrightarrow{AB} + k\overrightarrow{CD} \rightarrow$
$(XB - XA) = k(XD - XC) \wedge (YB - YA) = k(YD - YC)$

Let's consider the proof of the lemma *trans_parallel*: from parallelism of line AB and CD, we have existence of a real number k such that $\overrightarrow{AB} = k\overrightarrow{CD}$. The third above property gives us

$$k\overrightarrow{CD} \rightarrow (XB - XA) = k(XD - XC) \wedge (YB - YA) = k(YD - YC)$$

As a result, we have

$$(XB - XA)(YD - YC) = k(XD - XC)(YD - YC) = (XD - XC)(YB - YA) \quad \square$$

Now, let's consider the proof of the lemma *trans_perpendicular*: from perpendicularity of line AB and CD, we have $\overrightarrow{AB} \perp \overrightarrow{CD}$, hence $\overrightarrow{AB} \cdot \overrightarrow{CD} = 0$. Using the second property, we replace $\overrightarrow{AB}$ with $(XB - XA)\overrightarrow{OI} + (YB - YA)\overrightarrow{OJ}$. Similarly, we replace $\overrightarrow{CD}$ with $(XD - XC)\overrightarrow{OI} + (YD - YC)\overrightarrow{OJ}$. We have

$$((XB - XA)\overrightarrow{OI} + (YB - YA)\overrightarrow{OJ})((XD - XC)\overrightarrow{OI} + (YD - YC)\overrightarrow{OJ}) = 0$$

Once again, we use the tactic *RingScalarProduct* to simplify the formula and get
$(XB - XA)(XD - XC)\overrightarrow{OI} \cdot \overrightarrow{OI} + (YB - YA)(XD - XC)\overrightarrow{OJ} \cdot \overrightarrow{OI} +$
$(XB - XA)(YD - YC)\overrightarrow{OI} \cdot \overrightarrow{OJ} + (YB - YA)(YD - YC)\overrightarrow{OJ} \cdot \overrightarrow{OJ} = 0$

From the construction of the Cartesian coordinate system OIJ, we have $\overrightarrow{OI} \cdot \overrightarrow{OI} = 1$, $\overrightarrow{OI} \cdot \overrightarrow{OJ} = 0$, $\overrightarrow{OJ} \cdot \overrightarrow{OI} = 0$ and $\overrightarrow{OJ} \cdot \overrightarrow{OJ} = 1$. Replacing the left parts of these equation by the corresponding right parts, we get

$$(XB - XA)(XD - XC) + (YB - YA)(YD - YC) = 0 \quad \square$$

So, translating theorem statements into the algebraic form is easily performed. However, difficulties that we meet are how to treat conditions in the hypotheses. To ensure existence of geometric objects, non-degenerate conditions usually state that two points are different and two lines are not parallel

The conventional treatment of conditions of this kind in this method is performed by putting them into the proof obligation. For example, instead of proving *Goal* with the hypothesis $A <> B$, we prove the new goal $A = B \vee Goal$ without this hypothesis.

Suppose $Poly1 = 0$ and $Poly2 = 0$ are respectively algebraic forms of $A = B$ and *Goal*. So, the algebraic form of the new goal is $Poly1.Poly2 = 0$.

This treatment is applied for all the conditions. As a consequence, the algebraic form of the proof obligation becomes complex. This makes the method very slow.

In fact, there are conditions in the hypotheses that can be eliminated when translating the theorems statement into the algebraic form. It is because theorem statements in algebraic form implicity contain degenerate cases. There are conditions that are introduced in the proof obligation after the translation.

For example, in example 2, we have to prove $BC \parallel B'C'$. The statement of this example in our library needs the conditions $B \neq C$ and $B' \neq C'$ to ensure the existence of lines BC and B'C'.

Using the lemma *trans_parallel* to translate the proof obligation in the algebraic form, we have to prove that

$$((XB) - (XC)) * ((YB') - (YC')) = ((YB) - (YC)) * ((XB') - (XC'))$$

It is easy to find that this equality is still true for the cases $B = C$ and $B' = C'$. Therefore, adding them to the proof obligation is redundant.

However, it is really difficult to determine which conditions are superfluous after the translation. We do not have a technique for this yet.

4.5 Conclusion and future work

We present in this chapter the use of our library as a foundation to integrate other geometry systems. The axioms of Hilbert's and Tarski's systems for plane geometry are expressed and proved in our library. This, in a logical point of view, affirms the consistence of the formalization of geometry notions in our system with Hilbert's and Tarski's systems.

Integrating the area method and the method of Gröbner bases makes it possible to reuse the formalizations of these methods. As a result, users can switch between interactive proving and automatic proving. This has benefits from a geometry proving point of view. By using automatic proving for minor proof obligations that would lead to long and tedious proofs, users can concentrate on major reasoning steps. Users can use automatic proving to ensure the correctness of what they are going to prove. Besides, this integration offers users multiple geometry points of view about the same problem.

However, there is still some work to be done. Translating proof contexts of our library into ones of the area method, that we present in this chapter, is only for the case where problems are stated by constructions. This needs to be improved to deal with the other case where problems are stated in the form of predicates. In this direction, an

approach presented in [6] allows us to get geometric constructions from given predicates by constructing constraint matric. However, it is implemented in Java.

To integrate the method of Gröbner bases, treatment of non-degenerate conditions also needs to be improved.

In the future, we plan to integrate Wu's method that was formalized in Coq by J.D. Genevaux [22]. We also intend to formalize the notion of full angle. This will lay foundations for formalizing the full angle method in Coq.

Chapter 5

Dynamic Geometry Tool for Interactive Theorem Proving

5.1 Context

Nowadays, dynamic geometry software (DGS, also called interactive geometry software) plays an important role and has highly influenced mathematics education. It allows users to create and then manipulate geometric constructions, primarily in plane geometry. Users start with free points and construct new geometry objects with provided constructions like lines, intersection points, circle, Euclidean transformations, etc. By moving these free points, users can observe their influence on other constructed objects. Therefore, conjectures can be discovered.

There are numerous applications for interactive geometry on the market with a variety of features. Many of them are really used in classroom in many countries and have certain effects. However, a few of them deals with proof related features. Generally, they are classified into 2 categories[1].

- DGSs allowing users to automatically verify conjectures by using automatic theorem proving
- DGSs allowing users to construct proofs of conjectures

For the first category, proofs are generated by using automatic theorem proving which is usually based on efficient automatic proof methods such as the Gröbner bases method[33], Wu's method[10], the area method[11], and the full-angles method[11]. Some dynamic geometry softwares of this category are as follows

MMP/Geometer[20] and GeoExpert[50] are strong systems which implement Wu's method, the area method, the full-angles method (for GeoExpert), and especially the

[1]This classification is also presented in the PhD thesis of J. Narboux

deductive database method[13]. The last method relies on a set of basic rule, and allows to find out traditional proofs. GeoExpert allows users to visually understand each step in generated proof.

Geothms[46] with a web interface uses the GCLC prover which is based on the area method. With the support of a repository of geometry theorems, users can store theorem statements and their proofs.

Geometry Explorer[48] also implements the full-angles method. However, it can automatically generate novel diagrammatic proofs corresponding with reasoning used in geometry theorem proving. It is a good illustration for proofs.

For the second category, users can construct proofs step by step. We want to cite here remarkable software such as Chypre[5], Cabri-Euclide[35] and Geometrix[26]. Generally, dynamic geometry software of this category provides a set of basic rules and allows users to guide proofs by using the rules. There are some differences for manipulation in these tools.

For Geometrix, the teacher can create an exercise with an specification of the applicable rules. From the hypotheses of a created figure, system performs deductions with the set of rules, then generate conjectures. The teacher picks one to state the exercise. In turn, the student, at each proving step, can select a rule and see propositions that this rule can offer from the current proof context (hypotheses). These propositions are automatically calculated. The student chose propositions to add into the hypotheses. Once the proof obligation appears, the proof is finished.

For Chypre, the problem solving process is represented by a graph of facts. Once users add an assertion, the system automatically takes into account two types of elements: reasoning steps to deduce this assertion form existing ones, and other assertions that can be deduced form this newly one (by using the provided rules).

In the two systems above, automatic pre-calculations can make proving easier for users. However, users do not really participate in deduction, in particular in rules selection. Cabri-Euclide fills this gap by allowing users to organize their proof. When users introduce a statement, the system only verifies if this leads to a circular argument. When users apply a rule, the system verifies the necessary conditions that ensure the correctness of this deduction , and requests users to prove them.

All above systems realize their proof related feature by implementing their own theorem prover. We can see some drawbacks as follows:

- At the level of interaction with users: in the first category, tools automatically generate proofs hence users can not be involved in the proof process. In the second category, users can guide proofs. However, this guidance has to follow preconceived scenarios and interaction with geometric objects in the figure during proofs is limited.

- At the level of extension: their proof related features are difficult to extend. In fact, the theorem prover of a dynamic geometry software is often specifically implemented for the corresponding methods. Consequently, providing a new method leads to implementing a new prover completely, sometimes changing the underlying system (logic, algebraic).

- At the level of proof related capability: users can not combine different kinds of proofs. For example, proofs by deduction of a rule can not be used at the same time as algebraic calculations

In this chapter, we present a proof system extension to construct traditional proofs and interact directly with geometrical objects during these proofs. It is an interface for interactive geometry proving which is a combination of the Geogebra dynamic geometry software [23] with the Coq proof assistant. We think that using a proof assistant like Coq to implement theorem proving can remove the above drawbacks.

There are some developments that also aim at the combination of DGSs and proof assitants. First, we want to cite here the work developed by J. Narboux[39][38]. He developed a DGS named Geoproof, that provides the proof related feature with the support of Coq. This tool allows users to construct geometric objects and conjecture. Once users want to prove this conjecture, the theorem statement is translated to a proof context in the Coq proof assistant (including hypotheses and goal). Geoproof then launches the Coq IDE (as a separated editor or embedded editor) to prove this theorem with the area method [37]. However, this system provides only automatic proofs.

Another one which is also a combination of a DGS with Coq is GeoView[6], but this tool works in the opposite direction. It produces a diagram in a DGS, namely GeoPlan, from a theorem statement in Coq.

5.2 Introduction to Geogebra

Geogebra is a free dynamic geometry software for education in schools. It was started by Markus Hohenwarter in 2001 in his master and PhD work[23]. It is implemented in Java and thus available for multiple platforms. It provides a new kind of tool for mathematics by joining geometry, algebra and calculus. It received several international educational software awards and is applied in education at schools in different countries.

The Fig.5.1 is a screen shot of this tool. Like other dynamic geometry software, Geogebra provides basic geometric objects such as points, vectors, straight lines, circle, and more complex constructions such as midpoint, parallel lines, circumcircle, etc.

Users can draw geometric objects by selecting the corresponding icon in the tool bar, or typing command in the textbox at the bottom as well. Actions are performed in the

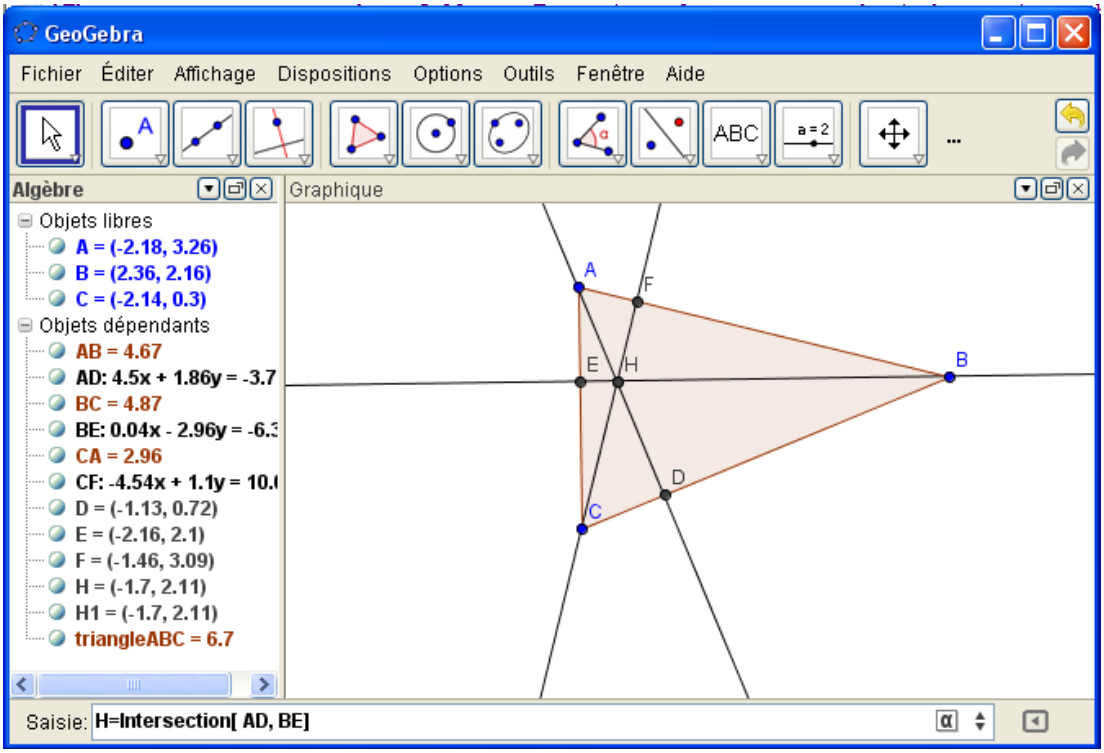


Figure 5.1: A screen shot of Geogebra

drawing window at the right. Geogebra also allows users to undo or redo constructions at anytime. Users than can move free points (A, B and C for the case of this figure) and observe the change of the others. By observing movements, users can find conjectures and check their correctness. For now, checks are limited to simple relations of two objects at the moment by using its underlying Computer Algebra System (CAS).

The left part of the figure is the algebraic window. It contain algebraic representations of geometric objects. When users move points in the drawing window, geometric constructions concerning these points are also changed. This change is automatically updated in the algebraic window.

Inversely, if users change values of coefficients of polynomials in the algebraic window, their geometric representation are changed.

This is a strong point of Geogebra which makes it different from other dynamic geometry software is the connection of Dynamic Geometry Software and Computer Algebra System. They are two types of educational software that connect the mathematical fields of geometry and algebra and are used for mathematics teaching and learning. However, they are usually treated in separated software packages.

Geogebra can profit both from visualization capabilities of computer algebra system and dynamic changeability of dynamic geometry system. This encourages students to discover mathematics in a bidirectional experimental way. Students can investigate

equations corresponding to drawn objects and they can also investigate the figures corresponding to given equations.

In fact, thanks to its advantages, Geogebra has a large community and used in classroom in many country. By combining Geogebra and Coq, we would like to provide a logical view for students on geometric problems besides what they have already.

5.3 Some analyses about school proofs

Let's return to example 1 mentioned in page 41. This is illustated in Fig. 5.2 and stated as follows: Let BD and CE be two altitudes of triangle ABC and points G and F be the midpoints of BC and DE respectively. It holds that GF $\perp$ DE.

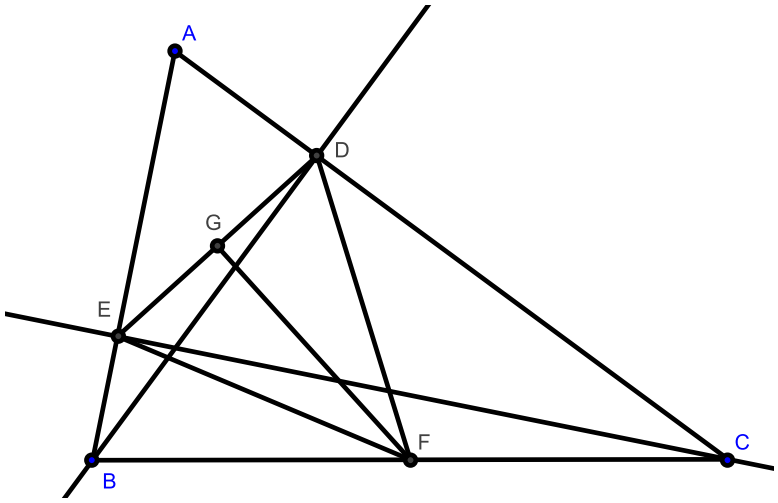


Figure 5.2: Example

We remind readers of the proof of this theorem. We use 2 following rules

Rule 1: Given a $\triangle MNP$ and I is middle point of NP. If $\triangle MNP$ is isosceles at M, then $MI \perp NP$.

Rule 2: Given a $\triangle MNP$ with a right angle in M, if I is middle point of NP, we have $|IM| = |IN| = |IP|$.

To prove $GF \perp DE$, by applying the first rule, we need to prove that $\triangle FDE$ is isosceles at F. In other words we have to prove $|FD| = |FE|$.

To prove this new goal. From the hypotheses of theorem, we have that $\triangle DBC$ a right triangle with A the right angle. This configuration satisfies rule 2. Using this rule gives us $|FD| = |FB| = |FC|$. By a similar way, we have $|FE| = |FB| = |FC|$. From the last two equations $|FD| = |FB| = |FC|$ and $|FE| = |FB| = |FC|$, we can easily prove that $|FE| = |FD|$.

There are three principal proof techniques used in proving geometry theorem : backward-chaining method, forward-chaining method and proving by drawing auxiliary

lines. These proof methods are accompanied with a set of basic rules.

The backward-chaining approach for geometry reasoning progresses from the conclusion to the hypotheses, i.e. we need to prove that

$\forall GeometricElements, Hyp_1 \wedge \cdots \wedge Hyp_n \rightarrow Goal$

We search in the set of base rule to find out a rule of the form

$\forall GeometricElements, G_1 \wedge \cdots \wedge G_n \rightarrow Goal$

Using this rule, the current goal is solved if the hypotheses of this rule are verified. So, the problem evolves into proving the new goal (also called subgoals) $G_1, \ldots, G_n$. This process is repeated for each subgoal until subgoals are in the hypotheses list or axioms.

The forward-chaining approach for geometry reasoning aims at generating new properties from the hypotheses by applying given rules. This means that if the hypotheses of a basic rule are in the hypotheses list of the proof context, then this rule can be used and its assertion is added in the proof context as a new hypothesis for the next deduction steps. In particular, if we need to prove that

$\forall GeometricElements, Hyp_1 \wedge \cdots \wedge Hyp_n \rightarrow Goal.$

And in the base of rules we have

$\forall GeometricElements, H_{j_1} \wedge \cdots \wedge H_{j_m} \rightarrow Goal_j$

with $\{H_{j_1}, \ldots, H_{j_m}\}$ are a subset of $\{Hyp_1, \ldots, Hyp_n\}$

So we can add G_j as a hypothesis of theorem. The process finishes when it can generate the goal.

Generally, we do not use only backward reasoning or forward reasoning to completely solve problems. These methods are alternatively used in deduction steps. In our example above,the backward reasoning is used in the deduction step where we apply rule 1. The proof obligation $GF \perp DE$ is replaced by 3 new goals: $FD = FE$, G is the middle point of DE, and FDE is a triangle (in other words, F, D and E are not collinear). However, the last subgoal that corresponds to a degenerate case is usually omitted in school proofs. The second subgoal is a hypothesis of theorem. So we only continue the proof by proving the first subgoal.

Forward reasoning is used in the next deduction step where we apply rule 2. Since we have $\triangle DBC$ with D the right angle and F is middle point of BC, this rule is applied and gives us new property $|FD| = |FB| = |FC|$.

The last method based on drawing auxiliary lines is used when the two methods above can not resolve problem. Auxiliary lines can make users to find out new useful properties. For example, this method is used in the proof of Ptolemy's theorem mentioned in Chapter 3.

5.4 The interface with the proof related feature

In this section, we present an extension of Geogebra with the proof related feature. This feature is performed by using our library mentioned in the previous chapter. We detail this system by showing how users can state and prove geometric theorems. Once again, example 1 is used for the purpose of explanation.

5.4.1 A snapshot of the interface

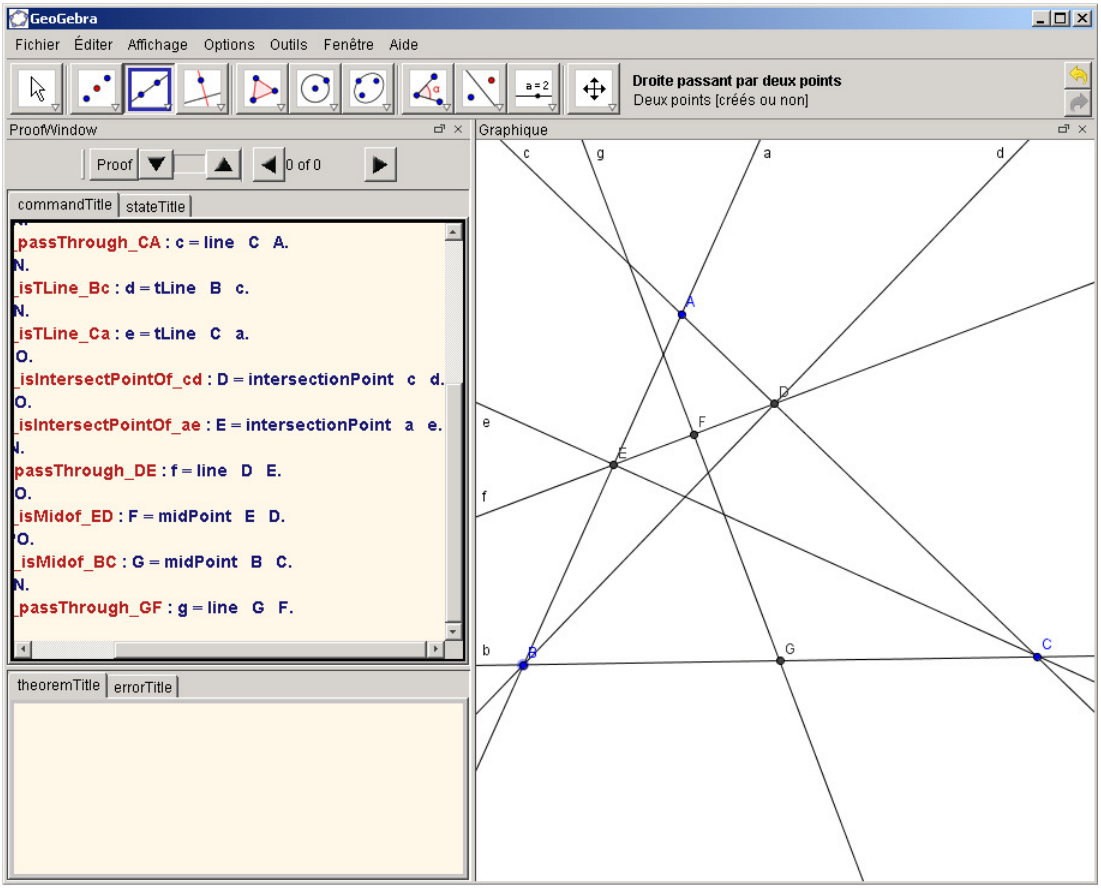


Figure 5.3: A screen-shot of GUI. A proof window is integrated in the left

The main interface is as in Fig. 5.3). A window is added for the proof related feature (the window in the left of the interface in the figure). It is used to interact with users in constructing proofs and contains two main sub-windows: the command window and the proof window. The command window is used to display all Coq commands which state and prove theorems. The proof window allows users to construct proofs, they can see hypotheses, and goals that need to be proved appear in this window. In addition, other sub windows are used to display applicable rules for a goal, to apply rules by giving values for their parameters, and to announce errors, etc.

Like other views, the proof window can be shown, hidden, dragged around to modify its position, and opened as an external window.

5.4.2 Stating theorems

5.4.2.1 Constructing diagrams

Geogebra provides a construction toolbar for users to construct diagrams. This toolbar contain common geometry constructions such as point, line, point on a line, the intersection point of lines, circle, triangle, etc. Geometry objects are created one by one by selecting appropriate constructions with the support of existing objects. During these step, users can undo and redo the last constructions that they create. Once the figure is drawn, users can move free points, and dependent constructions will be updated.

To construct a diagram corresponding to the above example, users draw a triangle $\triangle ABC$, construct perpendicular lines d $\perp$ AC such that B $\in$ d and e $\perp$ AB such that C $\in$ e, take intersection points E and D, and complete the figure by taking midpoints G and F of BC and DE respectively and joining them.

Each construction is translated into commands in Coq (figure 5.3). Adding new geometry objects or removing existing geometry objects implies adding or removing corresponding hypotheses.

Table 5.1: Constructive and predicate form of some constructions

Constructive form	Predicate form
M = midPoint (A, B)	collinear (A B M) $\wedge$ MA = MB
M = intersectPoint (l_1, l_2)	$l_1 \nparallel l_2 \wedge$ M $\in l_1 \wedge$ M $\in l_2$
l = line (A, B)	A $\neq$ B $\wedge$ A $\in$ l $\wedge$ B $\in$ l
l = parallelLine (A, m)	A $\in$ l $\wedge$ l $\parallel$ m
l = perpendicularLine (A, m)	A $\in$ l $\wedge$ l $\perp$ m

In most geometry theorem provers, the predicate form(right column of table 5.1) is used to describe geometric statements, i.e. hypotheses and conclusion are represented by geometry predicates. This offers a view about properties of geometric objects.

Our approach is different. We think that, in a pedagogical view, the constructive form(left column of table 5.1) is suitable for students during this phase. Thanks to the constructivity of our library, theorems are readily stated in this form.

However, the predicate form is necessary in proving theorems and users can obtain it in the proving phase which will be presented in the next section.

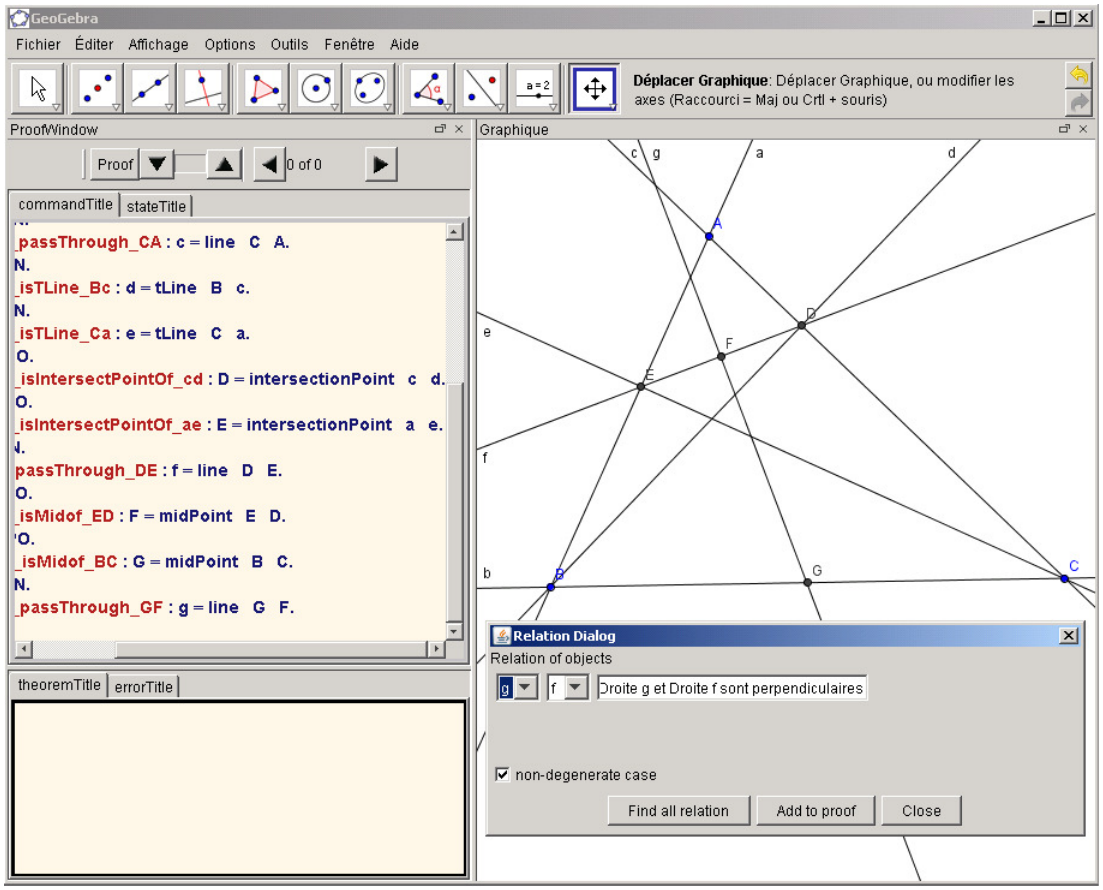


Figure 5.4: Users find and add a conclusion

5.4.2.2 Discovering conjectures

Once a diagram is completely constructed, users easily see free points that do not depend on other constructions as these points are highlighted using a different color. Users can move these free points by drag-and-drop, dependent constructions are updated, and users can observe relationships between geometry objects, traces of points, etc. and conjecture.

Once users want to prove a conjecture, they can check if this conjecture is correct. This is useful because they need to be ensure that they are going to prove a valid fact. This pre-verification can be performed using a feature of Geogebra with the support of its CAS, or using an automatic proof method in Coq (such as the area method).

For our example, users can move points A, B and C, they find out perpendicularity of GF and DE (GF $\perp$ DE). They select these lines and right click, a dialog appears to confirm this perpendicularity, and ask users to prove it (figure 5.4). Once users decide to prove the conjecture, we can go to theorem proving phase in the next section.

5.4.3 Theorem proving

As mentioned in Chapter 2, there is a big difference regarding the detail of proof between proofs in high school and formal proofs. In fact, at the cognitive level of high school students, they only concentrate on the main deduction steps that are performed by using the forward method and the backward method. Proving other subgoals that appear in the proof process does not adapt to their knowledge, hence these goals are usually omitted or admitted.

Consistent with proofs in school, our tool makes students concentrate on principal deductions steps. The auxiliary subgoals are solved by the underlying proving system, hence hidden from students' view.

5.4.3.1 Getting properties

We start this section by discovering geometric predicates from the construction. Our implementation allows users to get geometric predicates automatically or manually. Users can select a geometric object definition in hypotheses and get its properties using mouse-clicks, they can also unfold all definition in hypotheses at the same time. The corresponding tactics are automatically sent to Coq.

The following tactics collects hypotheses that define new points (for example M is the intersection point of lines a and b) and creates new hypotheses that are corresponding to properties of the new points (for example M $\in$ a, M $\in$ b and a $\nparallel$ b).

```
Ltac unfold_AllDefinition := match goal with
|H:?M = intersectionPoint ?a ?b |- _ =>
let H1 H2 H3 :=fresh "intersectionPointProperty" in
    destruct (@unfold_intersectionPoint M a b)
        as [H1 [H2 H3] ];auto;
    revert H; clear H; unfold_AllDefinition; intro H
end.
```

Once this tactic is applied, geometric object definitions are replaced by the corresponding predicates in the second column of table 5.1.

This form contains predicates such as $l_1 \nparallel l_2$, $A \neq B$, etc. which may be used as non degeneracy conditions for the existence of objects. With many definitions of geometric objects, we implicitly add non degeneracy conditions in hypotheses. These conditions are very important for reasoning. Resolving a problem in degenerate cases is often more complex than resolving the original problem. The same diagram with different paths of construction will have different non degeneracy conditions.

5.4.3.2 Searching and Applying rules

We present how the backward method is implemented in our system. Section 5.3 shows that for each backward deduction step, users need to find and apply a rule that leads to the current goal.

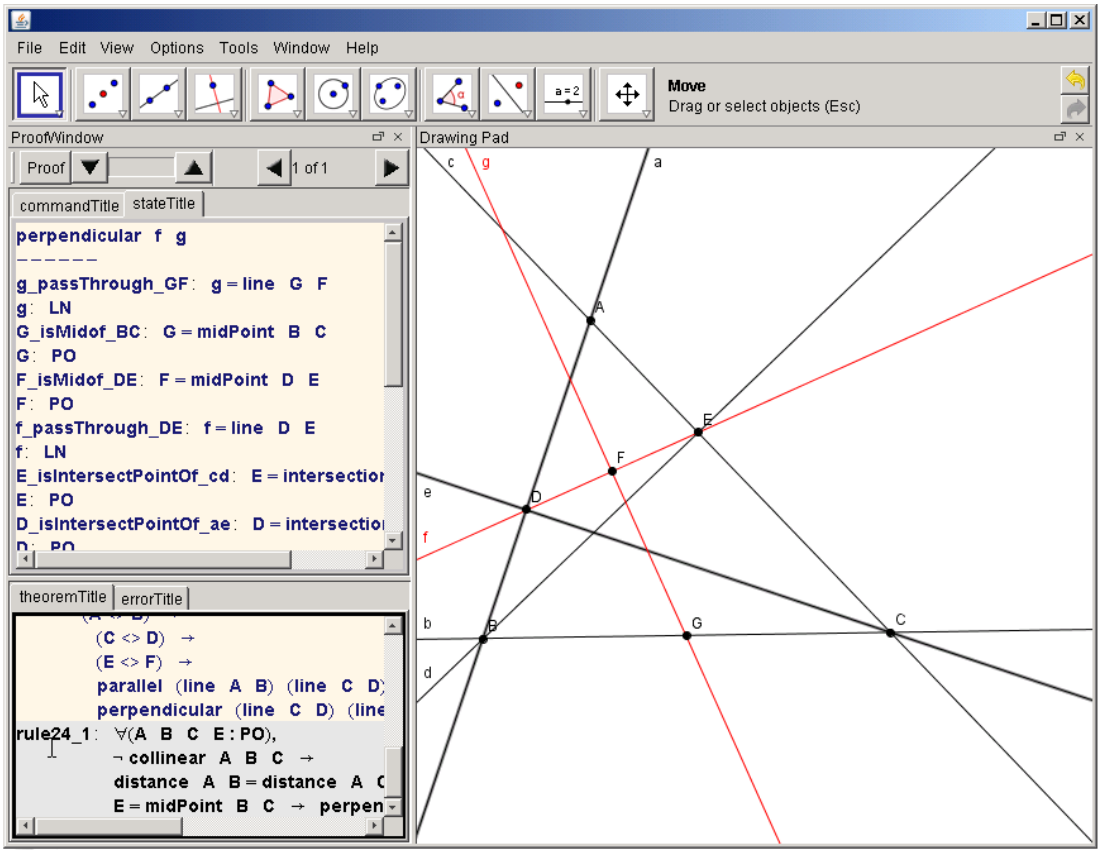


Figure 5.5: List of searched rules

Searching applicable rules in our interface is implemented using Search commands. It is strong enough to find all rules in a database which lead to a goal given by a pattern. From the proof window of the interface, users can search all applicable rules for the current goal. A list of applicable rules will be displayed (in the bottom of the proof window in Fig. 5.5). The result list is useful for users because it is a reduction of database of rules focusing in the current goal.

In our example 1, to prove that $GF \perp DE$, we need to find rules which have a conclusion in the form of perpendicularity of two lines. A variant of tactic *Search* is sent to Coq

SearchPattern ($\_\ \_ \perp \_\ \_$) *inside GeoBasicModule*

For another provided interaction users, we want to emphasize is the capability to visualize rules. Users not only have a list of applicable rules, but also can visualize

them. The corresponding figure of a rule makes it easier for users to select points for the application of this rule. In addition, it allows users to visually know what are the sub-goals generated by this application. The visualization of rules is illustrated in Fig. 5.6.

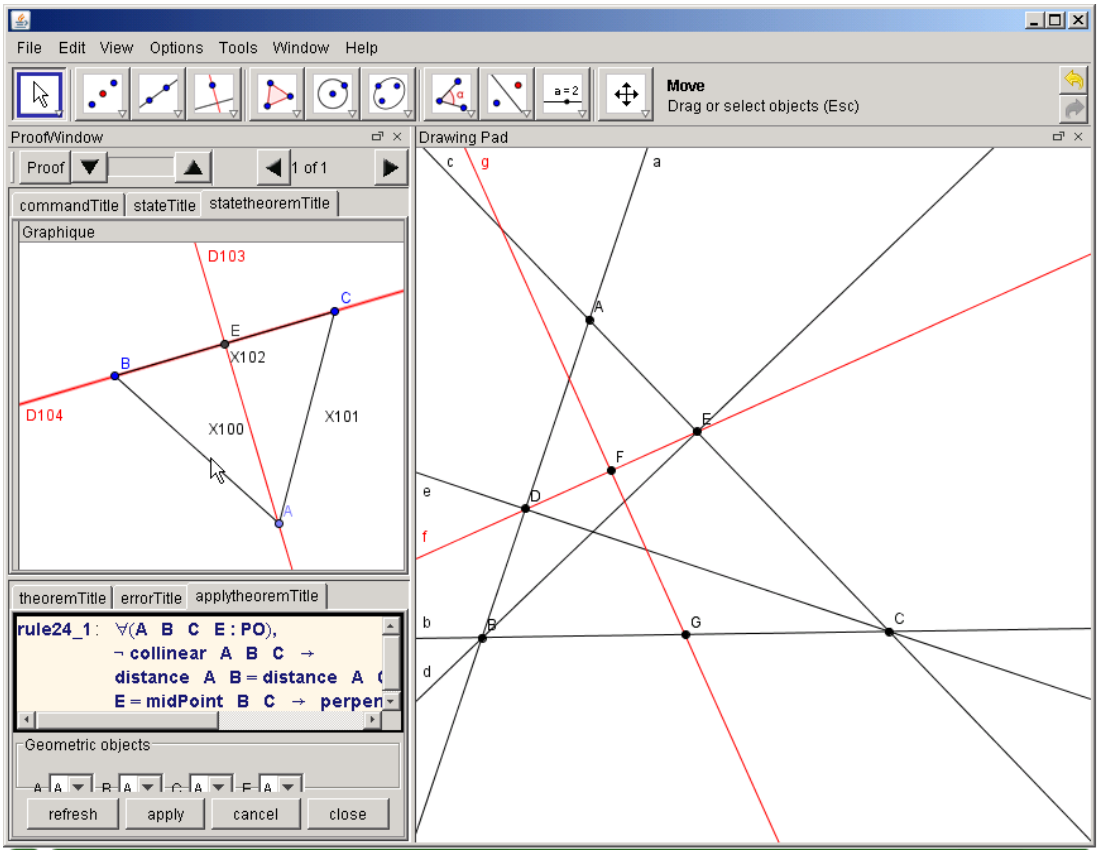


Figure 5.6: Corresponding figure of a rule

Users select a rule and view its corresponding figure. By moving points of this figure, they find out that there is a configuration of points in the theorem that satisfies this rule. After giving values for parameters of this rule which usually are points, lines, etc. users can update the statement of the rule with given values. Finally, users can apply this rule with assigned parameters to finish the deduction step.

In our example, the following rule is selected

Lemma rule24_1: $\forall$ A B C M :Point, B $\neq$ C $\rightarrow$ A $\neq$ M $\rightarrow$ AB = AC $\rightarrow$ M = MidPoint(B,C) $\rightarrow$ AM $\perp$ BC.

Users view its corresponding figure in the left window. By moving points, the figure is transform to the same configuration of points G, D, E and F. By assigning these points to corresponding parameters of the rule, users recognize that applying this rule gives $GF \perp DE$. The instance of rule is as follows

D $\neq$ E $\rightarrow$ G $\neq$ F $\rightarrow$ GD = GE $\rightarrow$ F = MidPoint(D,E) $\rightarrow$ GF $\perp$ DE.

Users can easily know which new goal they have to prove after this application $|GD| = |GE|$.

Applying this rule with selected points is performed by command

```
apply rule24_1 with (A:=G)(B:=D)(C:=E)(M:=F).
```

5.4.3.3 Adding new properties

Generally, we do not only use the backward method to solve problems. This method is used in alternation with the forward method. As mentioned, the forward method aims at generating new properties from hypotheses of the proof context by applying some rules. A rule can be applied when it is ensured in the proof context, in particular, hypotheses of the rule have to appear in the proof context.

This method is rather suitable to automatic deductions than interactive deductions, because applying this kind of method gives new properties without changing goals. However, sometimes in a proof, users can find new properties by moving free points in the figure and observing changes. These properties are useful for their proof. Unfortunately, these properties can not be directly deduced from hypotheses that they have. As a consequence, the forward method can not be applied. This case is a variant of this method and is provided in our system.

Our system, during proofs, allows users to add new properties that they find out from the figure into the proof context. As done with the conjecture of the theorem, users can select geometric objects, get properties among them and ask the system for a pre-verification. Then these properties are used as hypotheses in the current proofs once they are verified. This interaction is illustrated in Fig. 5.7.

In our example, users observe that $|GD| = |GC|$ and want to use this property. Users select 2 segments GD and GC, with a right click users can find out that they are equal and add this equation to the proof context. Proving these properties is performed as in the previous section.

In our example, $|GD| = |GC|$ is added into the proof context by the tactic *assert* and proved as follows

```
assert (distance G D = distance G C).
apply rule6_r with (A:=D)(B:=B)(C:=C)(E:=G).
```

Where *rule6_r* is stated by

```
Lemma rule6_r: ∀ A B C E :Point, A ≠ B → A ≠ C →
line(A,B) ⊥ line(A,C) → E = MidPoint(B,C) → EA = EC.
```

5.4.3.4 Drawing auxiliary lines

Sometime, users need to add lines, ray, segment, etc. to a diagram during their proofs. It is quite important to ensure the existence of these objects. In the theorem proving

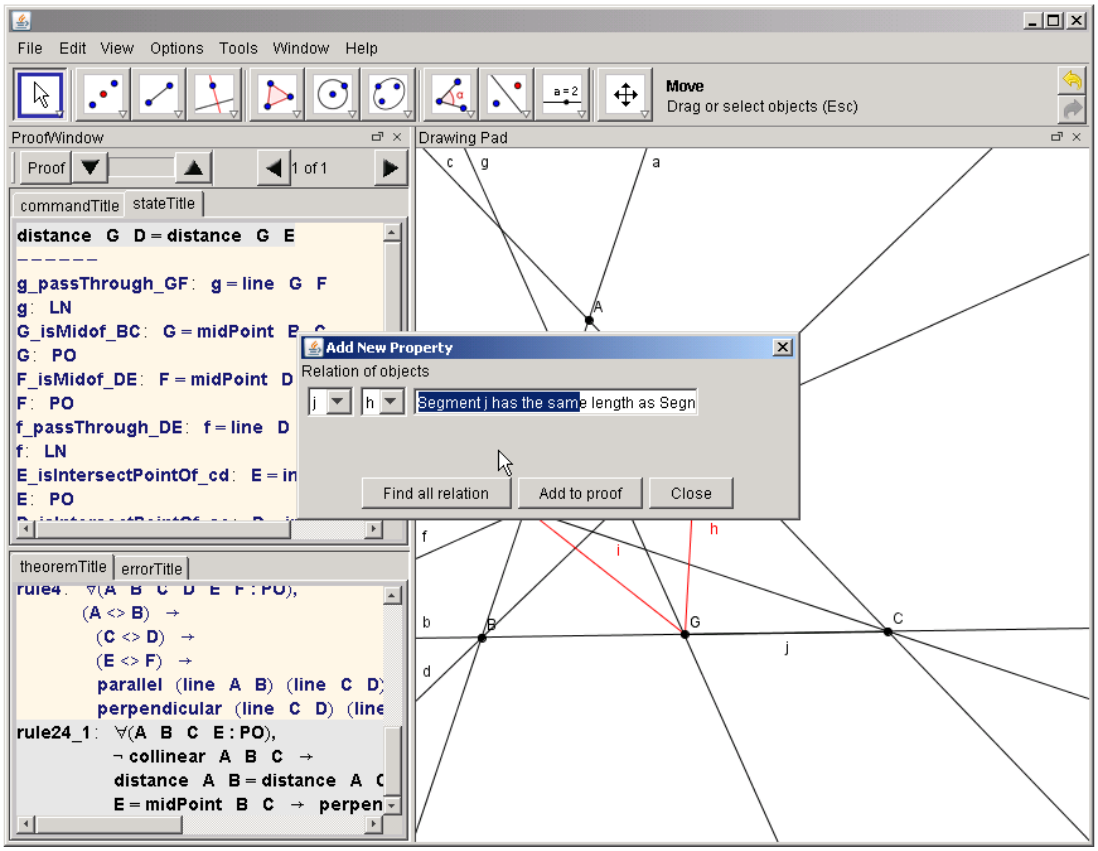


Figure 5.7: Corresponding figure of a rule

phase, each created object exists only under some conditions, the corresponding verification are sent to Coq. The following commands are sent to Coq when users create the intersection point M of 2 lines l_1 and l_2.

```
let H:= fresh "isIntersectionPoint" in
    assert(H: exists M, M = IntersectionPoint l1 l2;
    apply exists_intersectionPoint.
```

Coq requests to prove $l_1 \nparallel l_2$ to guarantee existence of M. Once this condition is verified, the point M will be displayed in the drawing window, and the command *destruct H as [M, H]* is automatically sent to Coq to have M = IntersectionPoint(l_1,l_2)) in hypotheses.

5.4.3.5 Automatic tactics

Proving geometry with the support of Coq gives a high level of confidence, each reasoning step is verified. However, it also complicates the process of proving. Users not only decide which rule will be applied but also prove many minor goals which lead to tedious proofs and is not suitable for a pedagogical setting. So, to avoid overwhelming users in proof details, we try to provide tactics to automatically solve these problems.

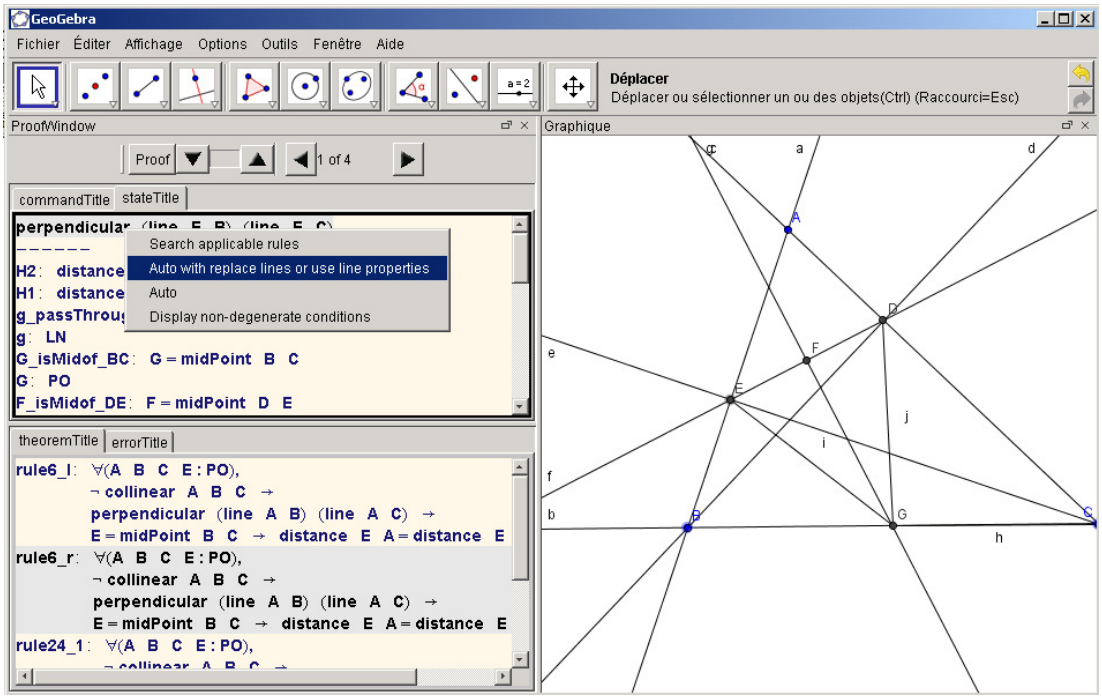


Figure 5.8: Users can use automatic tactics in popup menu

For example, we present a tactic to solve problems of line and points on the line. In our example, we have to prove that $line\ BD \perp line\ CD$ while we have $c = line\ AC$, $d = perpendicularLine\ B\ c$, $D = intersectionPoint\ c\ d$. With the tactic *unfold_AllDefinition* presented on page 5.4.3.1, we will get A ∈ c ∧ C ∈ c, B ∈ d ∧ d ⊥ c and D ∈ c ∧ D ∈ d. The following tactics try to replace all instance of line(M,N) by line l if it finds M ∈ l and N ∈ l in hypotheses.

```
Ltac replaceAllLineInstances := match goal with
|- context [line ?M ?N] =>
    match goal with | H1:liesOnLine M ?l |- _ =>
      match goal with | H2:liesOnLine M l |- _ =>
      try (replace (line(M,N)) with l in *;auto);
      replaceAllLineInstances
      end
    end
end.
```

So, one of our tactics is a combination of ”unfold_AllDefinition” and ”replaceAllLineInstances” tactics.

```
Ltac replace_auto_Lines :=
    solve[unfold_AllDefinition; replaceAllLineInstances].
```

For each particular goal, corresponding tactics are provided in a popup menu (figure 5.8).

5.5 Some discussion about proving with our interface

In comparing with proving at school, our system offers advantages in interaction with student and emphasizes logical reasoning in proofs. Students can understand hypotheses, goals, as well as rules that they use. The support of the system is moderate. Students themselves have to decide which rule is used, by which they can guide proofs. Corresponding figures of rules allows students to have a visual view of how rules are applied.

Besides, our system also provides some functions that make students more comfortable in proving. It can highlight geometric objects that students selected in hypotheses, by which students are clearer about correspondences between geometric objects and properties. It allows students to hide and display hypotheses. This avoid disturbing students with unimportant hypotheses.

In comparing with other DGSs with proof related features, interactions with users of our system are more familiar. Students feel free in proving by mouse clicks. We do not have scenarios for their proofs. They do proving in a similar way as they do in school. All support provided by our system aims at makes students easier to prove theorems but they do not lose the initiative in the proof process.

The base rules database of the system can be flexibly extended. In fact, it is because illustrated figures of rules are *automatically generated*, new rules are easily introduced in the database once they are proved and loaded. It also makes it easier for students to reused theorems that are proved by themselves.

Another advantage lies in the extension capability of theorem proving. In chapter 4, we presented combinations of our library with others automatic deduction methods such as the area method and the Gröbner basic method. These methods are implemented in Coq, and integrated in our library. This allows us to use alternatively these method in interactive proving.

5.6 Communication between Geogebra with Coq

Some integrated development environments (IDE) can be used to communicate with Coq. Here, we are interested in Pcoq which is a graphical user-interface for Coq [1]. Using Pcoq to communicate with Coq offers some benefits.

PCoq makes it possible to separate between the interface and the proof system. It means that the graphical interface and Coq are two independent processus. Using Pcoq with Geogebra allows users to continue manipulating geometric object while Coq performs deductions that users proposed.

Pcoq manipulates all formulas and commands as tree-like structures (also known as abstract syntax trees) rather than plain text. It means that commands sent to Coq

and responses received from Coq (including hypotheses and proof obligations) have tree structures and displayed by a structure editor. This allows us to easily access elements in proof context and also get information related to this elements. From this information, we can make our system to have reasonable interaction with users.

For example, if users select a hypotheses in the form of $a = line\ A\ B$, analyzing over its structure give us: this is relation with the operator "=", the left part is a variable namely a, the right part is a function namely *line* with 2 parameter A and B. Using these information, we highlight points A, B and line AB in the figure and allows users to use the tactic *rewrite* with this hypotheses.

Pcoq is implemented in Java, the same programming language as Geogebra, so this makes its integration in Geogebra easier.

5.7 Implementation

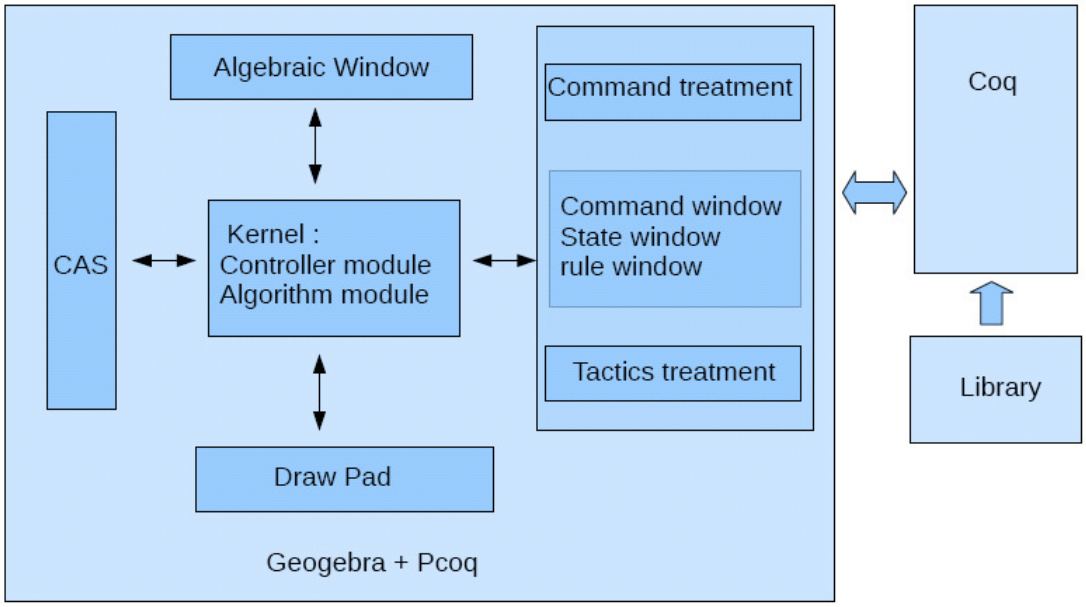


Figure 5.9: The system architecture.

An overview of our system is illustrated in Fig. 5.9. Integrating Pcoq into Geogebra makes it possible to communicate between Geogebra and Coq. In the section presenting the interface, we saw interactions between the draw pad and the proof view. In particular, when users draw geometry objects, corresponding command are sent to Coq to define these objects. We see also interactions between the proof view and Coq through sending commands and receiving proof contexts during proofs.

So, in this section, we present how we integrate Pcoq into Geogebra and clarify the implementation of its interactions with the draw window and Coq.

The version of Geogebra at the moment of concluding this work is 3.3.69.

5.7.1 Integration of the prove view

Geogebra is implemented in Java, its architecture is clear and well organized in separate layers and modules. It is built with a Model-Controller-View (so called MVC) model. The Model is application data, the View is a screen, and the Controller defines the way the View reacts to user input.

In Geogebra, we can simply understand that: the Model contains a list of built geometric constructions. The Controller contains functions to interact with users. For example when users create a line, Controller is responsible for creating a construction for a line and add it to the list in the Model. Finally, we have Views like the drawing view, or the algebra view to display information from the Model.

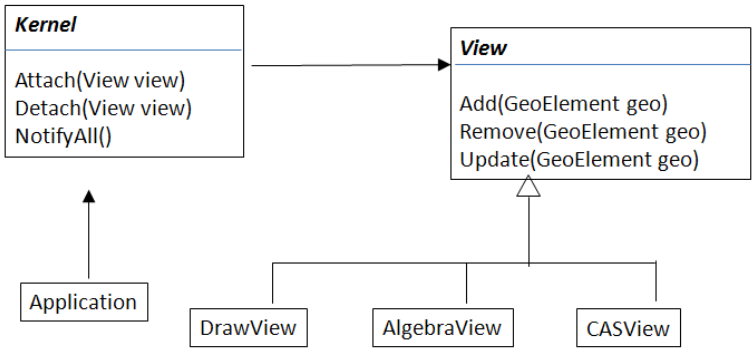


Figure 5.10: Implementing The Observer Pattern

Geogebra's views (including drawing pad, algebra view, CAS view, Spreadsheet view) are built in the MVC model by using the Observer pattern in Java (see Fig. 5.10). Therefore, they are synchronized each other and with the change of Model. For example, the representation of geometry objects in the drawing view and the algebra view are synchronized. Moving free points in the drawing view leads to a change of the construction list in the Model. This change is notified to all views by calling a function *update* with the changed construction. So, the algebraic representation of these objects are also changed.

Besides, they extend the JPanel class of Java. Thus, some features are easily provided that allow users to show or hide the views, drag around to modify their position, and open them in an external window as well.

Our integration of Pcoq in Geogebra respects this architecture. Pcoq is re-implemented as a view of Geogebra. This implementation makes Pcoq receive notifications from other views about changes of constructions. From this information, we can construct commands and send them to Coq.

5.7.2 Tree structure for geometric object definition

Now we consider creating objects in the phase of stating theorems. Once a geometrical object is drawn (for example the middle point of points A and B), a Java object of type *geometry element* is sent to all views. This Java object contains information such as the name of geometric object (suppose that M), the start point A, the endpoint B, and algorithm constructing this object (middle point). Our proof view receives this java object. After analyzing the contained information, we construct the corresponding hypotheses to define this geometry object. For example, the following code lines are for adding a midpoint in the phase of stating theorems:

```
private Tree[] commandMidPoint(GeoElement geo){
//this function is to produce commands for a midpoint
Tree[] contents=null;
AlgoElement parentAlgo = geo.getParentAlgorithm();
GeoElement point1 = parentAlgo.getInput()[0];
GeoElement point2 = parentAlgo.getInput()[1];
contents = new Tree[2];
contents[0] = TreeFormat.addVar(geo.getLabel(),
                                TreeFormat.Point);
contents[1] = TreeFormat.addHypothesis("Hyp_MidPoint",
    TreeFormat.assignVariable(geo.getLabel(),
        TreeFormat.function(TreeFormat.MidPointAB,
            point1.getLabel(), point2.getLabel())));
return contents;}
```

The function *commandMidPoint(GeoElement geo)* constructs a tree structure that declares a hypothesis with the name *Hyp_MidPoint*. The content of this hypothesis is the definition $M = midpoint\ A\ B$. Then this newly constructed command is sent to Coq as in the following code.

```
public class ProofView extends JPanel implements View {
 private GeoCommandManager gcManager;
 public void add(GeoElement geo) {
    gcManager.add(geo);
 }
}

public class GeoCommandManager {
 public Tree[] add(GeoElement geo) {
  Tree[] contents =null;
  AlgoElement parentAlgo = geo.getParentAlgorithm();
  if (parentAlgo instanceof AlgoMidpoint){
   contents = commandMidPoint(geo);
  }
```

```
    ..
    if (contents !=null){
     proofView.addTreeCommand(contents);//send command to Coq
    }
    return contents;
  }
}
```

The fact that we construct hypotheses in tree structures is also performed with command in the proving theorems phase. In the original version of Pcoq, users type commands in a plain text format. There is a parser to translate them to tree structures. However, we think that using an external parser is not secure. The correctness depends on the parser. We want to construct tree structure ourselves. This ensures that the right commands will be sent.

5.7.3 Interactions with users during their proofs

As mentioned, based on analyzing tree structure data, we can provide functions such as highlighting the selected objects, searching applicable rules, providing reasonable tactics and so on.

When users click on an element of a hypothesis, we can find the whole tree that expresses this hypothesis. Analyzing this tree allows us to figure out the semantics of this hypothesis and its elements. The grammar used for tree structures is called Vernac. The following is a part of this grammar

```
premise  -> ID FORMULA;
FORMULA  ::= appc prodc arrowc notation proj;
notation  -> STRING FORMULA_LIST;
FORMULA_LIST  ::= formula_list;
formula_list  -> FORMULA * ...;
appc  -> FORMULA FORMULA_NE_LIST;
```

A short description of this grammar is as follow : α -> A B means that a node of tree with operator name α (called node α) has 2 sub-nodes of type A and B; A :: α' β' A' means that a node type A can be implemented by nodes α', β', or other nodes of type A'; α -> A * .. means that node α has at least a sub-node.

As a result, the hypothesis $H : M = midpoint\ A\ B$ in the proof context has the tree structure given in Fig. **??**.

Our analyses in structures are performed by going down nodes. In particular, we check if a node is a premise. We go down the right branch and check if it is a notation. We then get names of the hypothesis (H), operator in this notation (operator +), used function and its arguments (midpoint, A and B). Implementation in Java is as follows:

```
if (selectedItem.operatorName().equalsIgnoreCase("premise")){
```

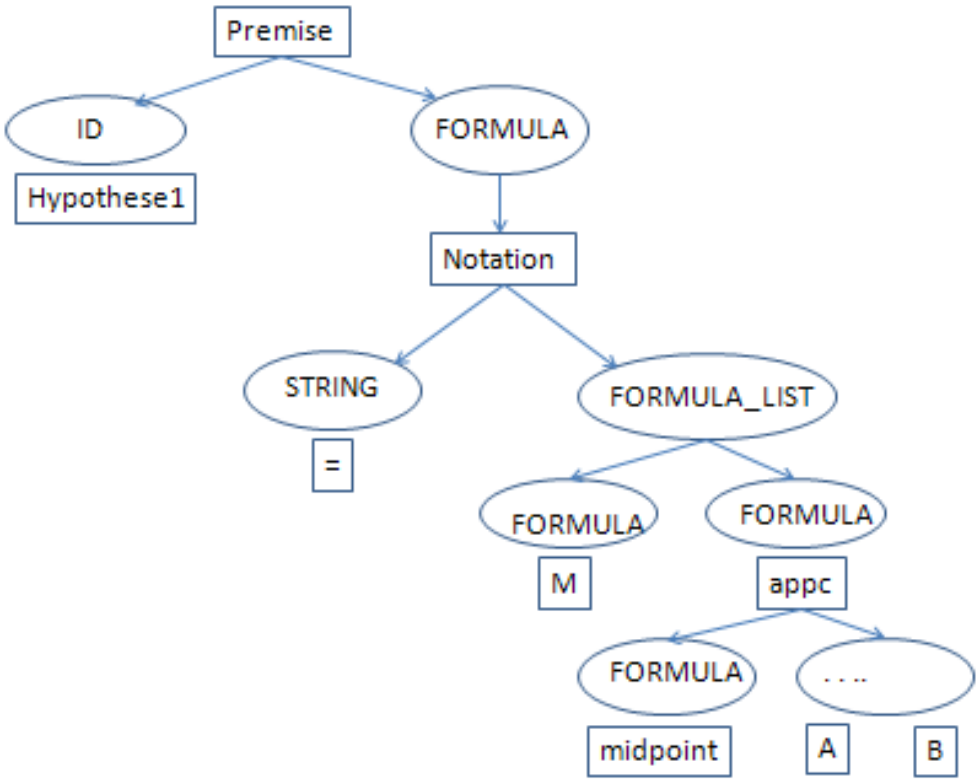


Figure 5.11: Tree structure of the hypothesis H: M = midpoint A B

```
 if (selectedItem.sons()[1].operatorName().
                equalsIgnoreCase("notation")){
  String hypName = selectedItem.sons()[0].atomValue().toString();
  Tree hypContent =selectedItem.sons()[1];
  String connecteur = hypContent.sons()[0].atomValue().toString();
  if (connecteur.equalsIgnoreCase("=")){
   miTmp = new MyMenuItem(ACTION_REWRITE_LEFT,
                          ACTION_NAME_REWRITE_LEFT,
                          MyMenuItem.NORMAL_TYPE);
   miList.add(miTmp);
   miTmp = new MyMenuItem(ACTION_UNFOLD,
                          ACTION_NAME_UNFOLD,
                          MyMenuItem.NORMAL_TYPE);
   miList.add(miTmp);
  }
...
}
```

This information allows us to provide reasonable assistance for users. In our code, we give users, in a popup menu, the tactic *rewrite* to replace M by its definition in the

proof context and the tactic *unfold* to get properties from this definition.

By providing assistances to allows users to guide precisely the proof process using mouse clicks, we continue the notion of "proof by pointing" that was approached by Y. Bertot *et al.* in Pcoq [7]. However, there are differences between these systems. While proof-by-pointing in Pcoq relies on analyzing formulas and understanding the meaning of logical symbols, our system focuses on geometric reasoning by getting geometric significations of formulas. In particular, with the hypothesis $H : M = midpoint\ A\ B$, the logical view only leads us to provide the tactic *rewrite*, whereas the geometrical view leads us to the tactics *rewrite* and *unfold_Definition*, the latter is to get properties from the definition.

5.7.4 Visualizing theorem statements

Figures of theorems, statements of theorems in the constructions form, statements of theorems in the predicate form are strictly related to each other. In the previous sections, we see that when we draw a figure, its geometric constructions are declared and these constructions are translated to predicates of geometric objects in the proving theorem phase.

We now consider an inverse direction of this relation which is how to obtain figures from predicates. As mentioned in the section 4.3.1, this is really difficult to implement in Coq, it is because there are probably several construction lists that lead to the same set of predicates.

However, this is implemented in Java based on calculations of a constraint set that are used in GeoView [6]. The idea of this implementation is as follows: from predicates we construct constraint set between points, then an algorithm is provided to produce constructions of points that satisfy constraints.

Reusing this algorithm, from a statement of a rule, we can obtain constructions of its points. By modifying the algorithm, we can receive constructions in its output under the forms of input command of Geogebra. It remains to instantiate the class in Java of the drawing view and use commands to produce the figure.

5.8 Conclusion and Future Work

In this chapter, we present our interface which allows users to construct interactively traditional proofs in geometry. The reasoning methods at high school level such as forward method, backward method, and drawing auxiliary objects as well are provided. Steps of reasoning are verified by a proof assistant hence the correctness is guaranteed.

Integrating a proof assistant in a dynamic geometry software offers a novel way of learning geometry. It allows users to additionally have a logic view in solving geometric problems. Users know what they have in hypotheses, what they have to prove, which

rules they can apply. By which, they can take good decisions and thoroughly understand reasoning steps.

Some features need to be improved to allow users to use our tool easily. Dynamically constructing a diagram from the statement of a rule allows users to better understand this rule. However, users still have to move free points manually in this constructed figure to find out points on the figure of the current theorem that conform to the rule. So the first improvement is how to automatically find out points for a selected rule.

Another improvement concerns the result list of searched rules. For now, users get rules by syntactic searching in Coq. Some of the rules obtained may be unapplicable in the current situation. We need to have a mechanism to eliminate rules of this kind. We could use the computer algebra system of Geogebra to do this

The last improvement we cite here is how we can organize and manage the set of applicable rules and the set of tactics while adapting to users levels. At this moment, all rules in our library can be used in proving. Of course, this is not suitable for all levels of students. Classifying them into levels is necessary.

Chapter 6

General Conclusion and Perpectives

6.1 Conclusion

In this thesis, we have presented our work about the formal description of geometric properties.

In Chapter 2, we described how to remove axioms in a development where axioms has been used for definitional purposes. We worked on a library that had been designed by F.Guilhot for its use by young students and focused in plane geometry.

Its axiomatic system was reduced to a system containing only axioms for mass point, scalar products and coplanarity. From this system, we built up affine geometry, then Euclidean geometry including trigonometry and planar transformation.

The number of axioms we rely on is 13, to which it should be added that we use the type of real number and all its properties.

A library was provided that covers almost all notions of plane geometry like lines, circles, intersection point of lines, parallelism of lines, perpendicularity of lines, collinearity of points, etc.

Geometry objects are defined in a constructive way, proofs are performed in a manner adapted to student's level of abstraction. It allows students to develop traditional geometric proofs as taught in school.

In Chapter 3, we described how new notions about orientation could be added to treat geometrical aspects that would otherwise be left implicit.

The notion of orientation was formalized by using a Cartesian coordinate system. Its close relationship with the notion of order was considered through proving order properties. We also clarified the role of orientation in formalizing other notions like oriented angles and similar triangles.

Our work provided a toolbox for orientation. This was tested by proving Ptolemy's

theorem and a theorem about product of segments of chords.

In Chapter 4, we described how axioms systems could be modeled in the library we developed and how automatic proving tool could be integrated.

Axioms in Hilbet's and Tarski's system for plane geometry were expressed and proved in our library. This, in a logical point of view, brings arguments for the consistence of our formalization of geometry notions with the other systems.

Axioms of the area method, a coordinate-free method, were verified. Tactics were provided to make it possible to translate geometric theorem statements in our library into one acceptable by this method. As a result, users can switch between interactive proving and automatic proving during proofs. Integration was also approached for the method of Gröbner bases that is an algebraic method for geometry.

With the fact that axioms systems were modeled and automatic methods were integrated, we provided a foundation for the formalization of geometry. Besides, we offered multiple geometry points of view about the same geometry problem.

In the last chapter, we presented a software tool that we developed which integrates the Coq proof assistant and the Geogebra dynamic geometry system. We also showed that many interface functionalities could be added to ease the work of developing formal proofs in geometry.

Using our software, users can interactively construct traditional proofs in geometry. Reasoning methods at school level such as forward method, backward method, and drawing auxiliary lines are provided.

Our software offers to users useful proof functionalities. At each deduction steps, users can request for applicable rules, users can see their illustrated figures by which users can give out good decisions. Moreover, direct interactions with geometric objects and reasonable assistance are provided during proofs making more natural proof processes.

6.2 Perspective

For the formalization in Coq, our main objective is to make proofs as natural proofs in school. This depends not only on which notions are provided in our library, but also on how we can manipulate them. Enriching the library by notions and properties need to be done.

Since there are a difference between formal proofs and school proofs in the level of their detail, we need to provide automatic tactics that is robust enough to solve minor deduction steps that appear in formal proofs and usually omitted in school proofs.

For our software, there are many ideas to improving this system. The first one lies on displaying contexts during proofs in natural language. In fact, in [6], the authors showed that statements in Coq can be expressed in natural languages with PCoq as

```
Lemma isosceles_median_bisector:
 (A, B, C, I : PO)
  ~ (A == I) ->
  ~ (B == C) ->
 (I == (midpoint B C)) ->
 (isosceles A B C) ->
 (cons_AV (vec A B) (vec A I)) == (cons_AV (vec A I) (vec A C)).
```

Lemma **isosceles_median_bisector**:
Let A, B, C and I be points.
If **A ≠ I,**
B ≠ C,
I is the midpoint of [BC],
and ABC is an isosceles triangle in A
then $(\overrightarrow{AB}, \overrightarrow{AI}) = (\overrightarrow{AI}, \overrightarrow{AC})$.

Figure 6.1: Statement in natural language

in Fig. 6.1. Therefore, this direction is reasonable. It makes the interface closer to students, hence it is a good direction from a pedagogical point of view.

The second one lies on showing proofs by using proof diagrams. Nothing could be better than using proof diagram to make students to understand proofs. This was also approached in Geometry Explorer[48] to display proofs developed by the full-angles method. With using Coq, we can re-use the tactic *Show Tree* that gives us a tree structure of our proof. This tactic is mentioned in [1] and proofs are expressed in natural language as in 6.2.

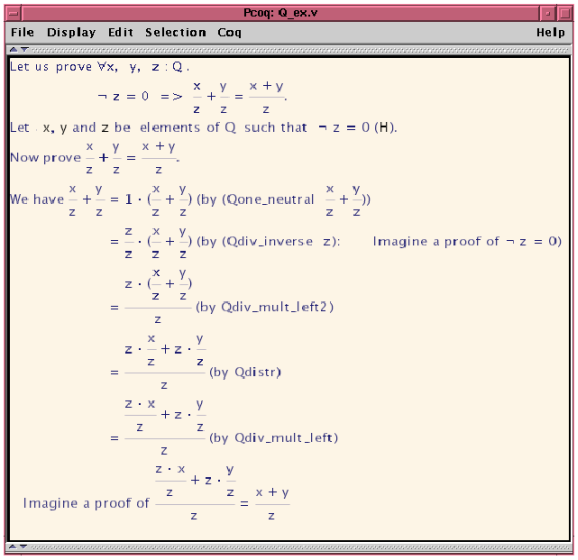


Figure 6.2: Proof in natural language

Instead of analyzing sent and received information between Geogebra and Coq during proofs, we can use this tactic to construct more easily proof diagrams. Moreover, thanks to using the structure editor of Pcoq, we can display proofs in natural language

as done with proof contexts.

By improving existing functionalities with adding these new functionalities into our system, we provide tools for students to perform integral proof processes.

Appendix A

Formal proof of a geometry theorem

This appendix contains the formal proof of the example 1. The theorem statement is as follows

```
Lemma perpExample: forall A B C D E F G :Point,
~col A B C ->
~ parallel_Line (lineT B (line A C)) (line A C)->
D = intersectionPoint (lineT B (line A C))(line A C) ->
~ parallel_Line (lineT C (line A B))(line A B) ->
E = intersectionPoint (lineT C (line A B))(line A B) ->
F = midpoint B C ->
G = midpoint E D ->
F<>G ->
D <> E ->
perpendicular (line F G)(line D E).
```

Two basic rules are used in the proofs

```
Lemma rule1_median_altitude : forall (M N P I :Point),
~col M N P -> I = midpoint N P -> distance M N = distance M P ->
perpendicular (line M I)(line N P).

Lemma rule2_median_rightTriangle : forall (M N P I :Point),
perpendicular (line M N) (line M P) ->
I = midpoint N P ->
distance I M = distance I N /\ distance I M = distance I P.
```

The proof is as follows:

```
intros.
(* Get properties from definitions of D and E *)
destruct (@plane_intersectionPoint_lyingOn
```

```
          D (lineT B (line A C))(line A C));auto.
assert (H10:=@lineT_property1 D B (line A C) H8).

destruct (@plane_intersectionPoint_lyingOn
          E (lineT C (line A B))(line A B));auto.
assert (H13:=@lineT_property1 E C (line A B) H11).

(* Prove some differences of points B<>D and C<>E *)
assert (B <> D).
red;intros.
rewrite <-H14 in *.
elim H.
lapply2 (@liesOnLineAB_col1 A C B);auto with geo.
unfold_triangle A B C;auto.
assert (C <> E).
red;intros.
rewrite <-H15 in *.
elim H.
lapply2 (@liesOnLineAB_col1 A B C);auto with geo.

(*Consider 2 cases D = C and D <> C*)
elim (classic (D = C));intros.
rewrite H16 in *.

(* For D = C, consider 2 cases E = B and E <> B*)
elim (classic (E = B));intros.
rewrite H17 in *.

(* 1st case D = C and E = B *)
(* Contradiction by AB // AC because both of them are
perpendicular with BC *)
assert (parallel_Line (line A B)( line A C)).
unfold_triangle A B C.
apply perp2_parallel with (line B C);auto with geo.
elim H;apply parallel_Line_col;auto.

(* 2nd case D = C and E <> B *)
lapply2 (@rule2_median_rightTriangle E B C F);auto with geo.
destruct H19.
apply (@rule1_median_altitude F C E G);auto with geo.
red;intros.
lapply2 (@col_eqDis_midpoint E C F);auto with geo.
rewrite <-H22 in *;intuition.
assert (line E B == line A B).
```

```
unfold_triangle A B C;
rewrite (@liesOnLine_lineEqual E B (line A B));auto with geo.
rewrite H18;auto with geo.

(* For D <> C, consider 2 cases E = B and E <> B*)
elim (classic (E = B));intros.
rewrite H17 in *.

(* 3rd case D <>C  and E = B *)
lapply2 (@rule2_median_rightTriangle D B C F);auto with geo.
destruct H19.
apply (@rule1_median_altitude F D B G);auto with geo.
red;intros.
lapply2 (@col_eqDis_midpoint B D F);auto with geo.
rewrite <-H22 in *;intuition.
assert (line D C == line A C).
unfold_triangle A B C;
rewrite (@liesOnLine_lineEqual D  C (line A C));auto with geo.
rewrite H18;auto with geo.

(* 3rd case D <>C  and E <> B, this is the non-degenerate case *)
lapply2 (@rule2_median_rightTriangle D B C F);auto with geo.
lapply2 (@rule2_median_rightTriangle E B C F);auto with geo.
apply (@rule1_median_altitude F D E G);auto with geo.
red;intros.
lapply2 (@col_eqDis_midpoint E D F);auto with geo.
rewrite <-H22 in *;intuition.
destruct H19;destruct H20; rewrite H19;auto.
destruct H19;destruct H20; rewrite H19;auto.

(* prove that EB and EC are perpendicular *)
assert (line E B == line A B).
unfold_triangle A B C;
rewrite (@liesOnLine_lineEqual E B (line A B));auto with geo.
rewrite H18;auto with geo.

(* prove that DB and DC are perpendicular *)
assert (line D C == line A C).
unfold_triangle A B C;
rewrite (@liesOnLine_lineEqual D  C (line A C));auto with geo.
rewrite H18;auto with geo.

Qed.
```

Appendix B

Axiomatic system

In this appendix, we present the axioms that we use in our system. This axiomatic system is more compact but enough to derive proofs of theorems in high-school scope. These axioms are suitable for the knowledge of students and are taught in school.

We preserve the approach of F. Guilhot in constructing Euclidean geometry from affine geometry because the notions of mass point and vector are presented in high-school courses, and calculations of mass point and vector are straightforward and familiar to students. The following axioms are for barycenter and mass point, which are used in her development:

Axiom 1 (Definition of addition) : $\forall$ (m n:Real)(P Q :Point)
$m + n \neq 0 \rightarrow \exists(R : Point), nP + mQ = (m + n)R$
Axiom 2 (Idempotency): nP + mP =(m+n)P.
Axiom 3 (Commutativity): nP + mQ = mQ + nP.
Axiom 4 (Associativity): nP + (mQ + kR) = (nP + mQ) + kR.
Axiom 5 (Definition of scalar multiplication): k(nP) = (k∗n)P.
Axiom 6 (Distributivity) k (nP + mQ) = knP + kmQ.

We use small letters a, b, c... to denote real numbers; capital letters A, B, C... to denote points; pairs of a real number and a point in the form aA to denote mass points

These axioms allow students to approach affine geometry through simple calculations of mass points. The notion of vector is then defined from mass points. To build up the Euclidean space, we introduce a system of axioms for the scalar product of vectors (denoted by $\overrightarrow{u} \cdot \overrightarrow{v}$), also called the dot product or the inner product. This notion is also familiar to students. It comes not only from mathematics, but also from physics.

Axiom 7 (positivity): $\forall$ $\overrightarrow{v}$:Vector, $\overrightarrow{v} \cdot \overrightarrow{v} \geqslant 0$
Axiom 8 (positivity2): $\forall$ $\overrightarrow{v}$:Vector, $\overrightarrow{v} \cdot \overrightarrow{v} = 0$ -> $\overrightarrow{v} = \overrightarrow{0}$
Axiom 9 (symmetry): $\forall$ $\overrightarrow{u}$ $\overrightarrow{v}$:Vector, $\overrightarrow{u} \cdot \overrightarrow{v} = \overrightarrow{v} \cdot \overrightarrow{u}$
Axiom 10 (additivity): $\forall$ $\overrightarrow{v_1}$ $\overrightarrow{v_2}$ $\overrightarrow{v_3}$:Vector, $(\overrightarrow{v_1} + \overrightarrow{v_2}) \cdot \overrightarrow{v_3} = \overrightarrow{v_1} \cdot \overrightarrow{v_3} + \overrightarrow{v_2} \cdot \overrightarrow{v_3}$
Axiom 11 (homogeneity): $\forall$ (k : R)($\overrightarrow{u}$ $\overrightarrow{v}$:Vector), $(k \times \overrightarrow{u}) \cdot \overrightarrow{v} = k \times (\overrightarrow{u} \cdot \overrightarrow{v})$

Finally, to define the Euclidean plane, the following axioms are added, they assert the existence of 3 non-collinear points and their co-planarity with any fourth point.

Axiom 12 (existence of 3 not aligned points): there are 3 different and non−aligned points O, O_1 and O_2.
Axiom 13 (coplanarity): **for** any 4 points A, B, C and D in the plane, we alway have $\overrightarrow{AD}$ is linear combination of $\overrightarrow{AB}$ and $\overrightarrow{AC}$.

Note that we here give the axioms in their mathematic forms.

Bibliography

[1] Ahmed Amerkad, Yves Bertot, Loïc Pottier, and Laurence Rideau. Mathematics and Proof Presentation in Pcoq. Technical Report RR-4313, INRIA, November 2001. 114, 124

[2] Jeremy Avigad, Edward Dean, and John Mumma. A formal system for euclid's elements. *Review of Symbolic Logic*, 2009. 49

[3] Michael T Battista and Douglas H Clements. Geometry and proof. *National Council of Teachers of Mathematics*, 88(1):48–54, 1995. 1

[4] Marcel Berger, P. Pansu, J.-P. Berry, and X. Saint-Raymond. *Problems in Geometry*. Springer, 1984. 8

[5] Philippe Bernat. Chypre: Un logiciel d'aide au raisonnement. Technical Report 10, IREM, 1993. 100

[6] Yves Bertot, Frédérique Guilhot, and Loïc Pottier. Visualizing geometrical statements with geoview. *Electron. Notes Theor. Comput. Sci.*, 103:49–65, November 2004. 98, 101, 120, 123

[7] Yves Bertot, Gilles Kahn, and Laurent Théry. Proof by pointing. In *TACS*, pages 141–160, 1994. 120

[8] Bruno Buchberger and Franz Winkler. *Gröbner bases and applications*. Cambridge University Press, 1998. 48

[9] Shang-Ching Chou. *Proving and discovering geometry theorems using Wu's method*. PhD thesis, The University of Texas, Austin, December 1985. 48

[10] Shang-Ching Chou. *Mechanical Geometry Theorem Proving*. D. Reidel Publishing Company, 1988. 99

[11] Shang-Ching Chou, Xiao-Shan Gao, and Jing-Zhong Zhang. *Machine Proofs in Geometry*. World Scientific, Singapore, 1994. 48, 82, 99

[12] Shang-Ching Chou, Xiao-Shan Gao, and Jing-Zhong Zhang. Automated generation of readable proofs with geometric invariants, theorem proving with full angle. *Journal of Automated Reasoning*, 17:325–347, 1996. 48, 70

[13] Shang-Ching Chou, Xiao-Shan Gao, and Jing-Zhong Zhang. A deductive database approach to automated geometry theorem proving and discovering. *J. Autom. Reasoning*, 25(3):219–246, 2000. 100

[14] Pichardie David and Bertot Yves. Formalizing convex hulls algorithms. In *Proc. of 14th International Conference on Theorem Proving in Higher Order Logics (TPHOLs'01)*, volume 2152 of *Lecture Notes in Computer Science*, pages 346–361. Springer-Verlag, 2001. 49, 54

[15] M. De Villiers. Using dynamic geometry to expand mathematics teachers understanding of proof. *International Journal of Mathematical Education in Science and Technology*, 35(5):703724, 2004. 3

[16] Christophe Dehlinger, Jean-François Dufourd, and Pascal Schreck. Higher-order intuitionistic formalization and proofs in Hilbert's elementary geometry. In *Automated Deduction in Geometry*, pages 306–324, 2000. 6, 72

[17] Jean Duprat. Constructors: a ruler and a pair of compasses. Types 2002, April 2002. 72

[18] Jean Duprat. Une axiomatique de la géométrie plane en coq. In *Actes des JFLA 2008*, pages 123–136. INRIA, 2008. In french. 49

[19] Freek Wiedijk. . Formalizing 100 theorems. `www.cs.ru.nl/~freek/100/`. 67

[20] Xiao-Shan Gao and Qiang Lin. Mmp/geometer - a software package for automated geometric reasoning. In *Automated Deduction in Geometry*, pages 44–66, 2002. 99

[21] Thomas Gawlick. Connecting arguments to actions dynamic geometry as means for the attainment of higher van hiele levels. *ZDM*, 37(5):361–370, 2005. 3

[22] Jean-David Genevaux, Julien Narboux, and Pascal Schreck. Formalization of Wu's simple method in Coq. In Jean-Pierre Jouannaud and Zhong Shao, editors, *CPP 2011 First International Conference on Certified Programs and Proofs*, LNCS, Kenting, Taiwan, Province Of China, December 2011. Springer-Verlag. 98

[23] Geogebra development team. Introduction to geogebra. `http://www.geogebra.org/book/intro-en/`. 101

[24] Marvin Jay Greenberg. *Euclidean and Non-Euclidean Geometries: Development and History*. W. H. Freeman; 4th edition, 2007. 6

[25] Benjamin Grégoire, Loïc Pottier, and Laurent Théry. Proof Certificates for Algebra and their Application to Automatic Geometry Theorem Proving. *LNAI*, 2010. 95

[26] Jacques Gressier. Geometrix, 1988-1998. http://perso.wanadoo.fr/jgressier/ENGLISH/english.html. 100

[27] Frédérique Guilhot. Premiers pas vers un cours de géométrie en Coq pour le lycée. Technical report, INRIA, July 2003. 7, 8, 15

[28] Frédérique Guilhot. Formalisation en coq d'un cours de géométrie pour le lycée. In *Journées Francophones des Langages Applicatifs*, Janvier 2004. 7, 8, 15

[29] Robin Hartshorne. *Geometry: Euclid and beyond*. Springer, 2000. 6

[30] David Hilbert. *Les fondements de la géométrie*. Dunod, Paris, Jacques Gabay edition, 1971. Edition critique avec introduction et compléments préparée par Paul Rossier. 6, 73

[31] Predrag Janičić. Geometry construction language. *Journal of Automated Reasoning*, 44:3–24, 2010. 12

[32] Predrag Janicic, Julien Narboux, and Pedro Quaresma. The Area Method : a Recapitulation. *Journal of Automated Reasoning*, 2010. 89

[33] Deepak Kapur. Using groebner bases to reason about geometry problems. *J. Symb. Comput.*, 2:399–408, December 1986. 72, 99

[34] Donald Knuth. Axioms and hull. *Lecture Notes in Computer Science*, 1991. 49, 50

[35] Vanda Luengo. *Cabri-Euclide: Un micromonde de Preuve intégrant la réfutation*. PhD thesis, Université Joseph Fourier, 1997. 100

[36] Laura Meikle and Jacques Fleuriot. Formalizing Hilbert's Grundlagen in Isabelle/Isar. In *Theorem Proving in Higher Order Logics*, pages 319–334, 2003. 7

[37] Julien Narboux. A decision procedure for geometry in Coq. In *TPHOLs'04*, volume 3223 of *LNCS*, pages 225–240. Springer-Verlag, 2004. 72, 101

[38] Julien Narboux. *The user manual of GeoProof*, July 2006. http://home.gna.org/geoproof/documentation.html. 101

[39] Julien Narboux. A graphical user interface for formal proofs in geometry. *J. Autom. Reasoning*, 39(2):161–180, 2007. 101

[40] Julien Narboux. Mechanical theorem proving in Tarski's geometry. In *ADG'06*, volume 4869 of *LNAI*, pages 139–156. Springer-Verlag, 2007. 7, 72

[41] Julien Narboux. Mechanical theorem proving in Tarski's geometry. In *Post-proceedings of Automatic Deduction in Geometry 06*, volume 4869 of *LNCS*, pages 139–156. Springer, 2008. 72, 73, 81, 83

[42] Tuan Minh Pham. Similar Triangles and Orientation in Plane Elementary Geometry for Coq-based Proofs. *SAC '10 Proceedings of the 2010 ACM Symposium on Applied Computing*, March 2010. 5

[43] Tuan Minh Pham and Yves Bertot. A combination of a dynamic geometry software with a proof assistant for interactive formal proofs. In *9th International Workshop On User Interfaces for Theorem Provers FLOC'10 Satellite Workshop*, Electronic Notes in Theoretical Computer Science (ENTCS), Edinburgh, Scotland, Royaume-Uni, 2010. Elsevier. 5

[44] Tuan Minh Pham, Yves Bertot, and Julien Narboux. A Coq-based Library for Interactive and Automated Theorem Proving in Plane Geometry. In *The 11th International Conference on Computational Science and Its Applications (ICCSA 2011)*, volume 6785 of *Lecture Notes in Computer Science*, pages 368–383, Santander, Spain, 2011. Springer-Verlag. 5

[45] Loïc Pottier. Connecting gröbner bases programs with coq to do proofs in algebra, geometry and arithmetics. In *Proceedings of the Combined KEAPPA - IWIL Workshops*, pages 67–76, 2008. 72

[46] Pedro Quaresma and Predrag Janičić. Geothms — a web system for euclidean constructive geometry. *Electron. Notes Theor. Comput. Sci.*, 174:35–48, May 2007. 100

[47] Wolfram Schwabhauser, Wanda Szmielew, and Alfred Tarski. *Metamathematische Methoden in der Geometrie*. Springer-Verlag, 1983. In german. 7, 72

[48] Wilson Sean and Fleuriot Jacques. Geometry explorer: Combining dynamic geometry, automated geometry theorem proving and diagrammatic proofs. *Proceedings of the 12th Workshop on Automated Reasoning (ARW)*, 2005. 100, 124

[49] Marshall Stone. Learning and teaching axiomatic geometry. *Educational Studies in Mathematics*, 4(1):91–103, 1971. 7

[50] Xiao-Shan Gao. Geometry expert, software package. `http://www.mmrc.iss.ac.cn/~xgao/gex.html`. 99