Internship at the Technische Universität Berlin

# Proving Statements in Planar Geometry
and
# *Cinderella*

Dan Roozemond - 0492344
Eindhoven University of Technology
Department of Mathematics and Computer Science
Discrete Mathematics and its Applications

July - October, 2003

## Abstract

This report describes the things done in a three month internship at the Technische Universität Berlin, performed in the summer of 2003. The main goal of this internship was to make it possible to put forward a geometrical theorem (i.e. a theorem involving points, lines, conics, incidences, etc) on a computer by pointing and clicking, and then obtain a computer-generated proof for this theorem.

This goal was achieved, adding various options to the computer program *Cinderella*[4]. With Cinderella one can create geometrical configurations, and the internal 'Randomized prover' is able to discover theorems. In this internship we added the functionality to find proofs for these theorems with the aid of the computer algebra package *GAP*[20]. Communication between these two programs and the various steps in generating the proof is done by means of *OpenMath* [12, 15].

Cinderella is now able to generate a mathematically correct proof of certain theorems created by the user. Moreover, one can verify this proof without extensive knowledge of the way the proof was obtained. As this internship was performed as part of the Masters education 'Discrete Mathematics and its Applications,' this report focuses on the mathematics behind the software rather than on the software itself.

The classic way to automatically prove geometric theorems is via translation into polynomials, so a thorough explanation of the mathematics involved in doing so is given in Chapter 3. It then will be shown that the classic way is in this case simply not the way to go. It will become clear that Gröbner bases are not fit for proving geometric theorems when the translation into polynomials has to be done by a computer program. A much faster, but less powerful, alternative is presented in Chapter 6. This alternative, based on bracket calculations as proposed in [16], was implemented as an add-on for Cinderella. This functionality originally initiated the development of Cinderella, but was lost five years ago. Moreover, the prover made in this internship has the advantage that it uses OpenMath to communicate with Cinderella. In Chapter 9 some examples of the prover in action are given.

# Contents

# Chapter 1

# Introduction

*Proof is the idol before whom the pure mathematician tortures himself.*
– Sir Arthur Eddington (1882 - 1944)

The internship covered in this report was conducted in the scope of *Automatic Geometric Theorem Proving*. The main goal was to make it possible to put forward a theorem by pointing and clicking, and then obtain a mathematically sound proof of that theorem. This goal was achieved by using the following four packages:

**Cinderella** "Software for doing geometry on the computer, designed to be both mathematically robust and easy to use" [4].

**OpenMath** "A new, extensible standard for representing the semantics of mathematical objects" [12] - The communication between Cinderella, SINGULAR and GAP was implemented with OpenMath. This was done using the Riaca OpenMath library for Java [15].

SINGULAR "A Computer Algebra System for Polynomial Computations" [6].

**GAP** "GAP – Groups, Algorithms, and Programming" [20].

It was clear that the layout of the project had to conform to Figure 1.1. Although the first step may seem (and in the case of Cinderella actual *is*) a rather trivial one, we do require it. By this, we explicitly disconnect the program creating the geometric theorem and the program proving that theorem. On the one hand, this enables us to try different provers without having to change the way Cinderella outputs its configuration, on the other hand the prover could easily import geometric theorems created in other packages.

The initial layout used can be found in Figure 1.2. In a Bachelor's project I had been working on in the spring of 2003 [18], steps 3 and 4 were realized. Having solved this part of the problem, it looked like the main thing that had to be done in this internship was the translation from objects in Cinderella (lines, points, etc) to polynomials, i.e. Steps 1 and 2 from Figure 1.2. As simple as this may seem, the question remains how to do this in such a way that SINGULAR is able to obtain an answer in a reasonable amount of time. This question is not easily answered,

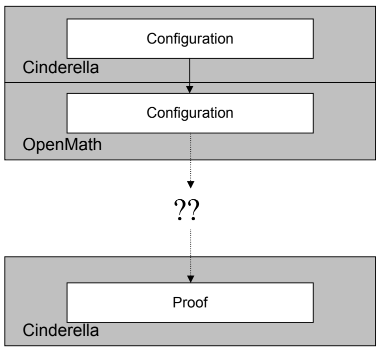

Figure 1.1: The template

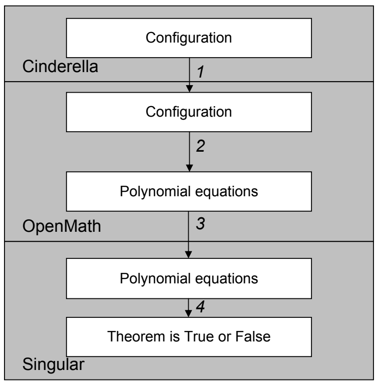

Figure 1.2: Initial layout

although there has been a lot of research on this topic. It will become clear that Gröbner bases are not made for proving statements put together by a user, and translated to polynomial equations by a computer program.

This report starts out with a description of how mathematical objects like points and lines are represented in Cinderella. Chapters 3 and 4 focus on the well-known and well-researched field of proving statements in planar geometry using ideals, Hilbert's Nullstellensatz and Gröbner bases. In Chapter 5 it will become clear why this methods are not suitable for proving statements created in Cinderella.

In Chapter 6 a rather different method (using *bracket algebra*) is proposed, that certainly has some drawbacks compared to Gröbner bases, but has the advantage of normally completing its calculations within the lifetime of the person asking for it. Moreover, the disadvantages are not as big as they might seem, as shown in Chapter 7. This is the method implemented and incorporated in Cinderella, in Chapter 8 some notes on this implementation are given. Finally, Chapter 9 shows some examples of theorems that were proved using this implementation.

Whenever one of the words 'he', 'his' or 'him' is used in this report you can try to replace it by one of 'she' or 'her'. If the sentence still makes sense, I meant to include both versions.

*Dan Roozemond, October 2003*

Chapter 2

# Mathematics in Cinderella

In this chapter it is tried to make clear how Cinderella handles mathematics, how geometry is represented, etc. It is clear that this knowledge is required to successfully conduct this project. The information presented here is taken from [9].

## 2.1 Foundations of Dynamic Geometry

In [9, Chapter 4] a formal framework for Dynamic Geometry is presented. One of the most important notions is the *Relational Instruction Set*:

**Definition 2.1.1. (Relational Instruction Set (RIS))** A *relational instruction set* is a pair $(O, \Omega)$ of *objects* O and *primitive operations* (or *primitives*) Ω with the following properties:

1. $O = (O_1, \ldots, O_k)$ is a family of sets $O_i$. These sets partition the objects into classes of the same type.

2. The primitive operations in Ω are relations

$$\omega_i \subset (O_{x_1} \times \ldots \times O_{x_{s_i}}) \times O_{x_{s_i+1}}$$

with input size $ar(\omega_i) = s_i$. An element of $(O_{x_1} \times \ldots \times O_{x_{s_i}})$ is called *input*, $O_{x_{s_i+1}}$ is called *output* of $\omega_i$

This definition is clarified by the following example:

**Example (Projective Geometry)** For a projective plane $\mathbb{P}$ of points $P$ and lines $L$ let

**Join** = $\omega_1$ := $\{(p_1, p_2, l)$ such that $l$ is the line through $p_1$ and $p_2 \neq p_1\}$

**Meet** = $\omega_2$ := $\{(l_1, l_2, p)$ such that $p$ is the point on $l_1$ and $l_2 \neq l_1\}$

Then $(\omega_1 \subset (P \times P) \times L$ and $\omega_2 \subset (L \times L) \times P$, and $\Omega = (\omega_1, \omega_2)$, $((O_1 = P, O_2 = L), (\omega_1, \omega_2))$ is a RIS describing the notions 'meet' and 'join' in Projective Geometry. The objects are either a point or a line, all (both) primitives have input size 2 ($s_1 = s_2 = 2$). Furthermore, this RIS is determined, i.e. for a given input there is at most one possible output.

**Definition 2.1.2. (Geometric Straight-Line Program)** A *geometric straight-line program* or *GSP* on a relational instruction set $(O, \Omega)$ is defined by a triple $(X, R, \Gamma)$:

1. $X = (X_1, \ldots, X_n)$ are called *input variables,*
2. $R = (R_0, \ldots, R_{m-1})$ are called *output variables* or *intermediate variables,*
3. $\Gamma = (\Gamma_0, \ldots, \Gamma_{m-1})$ are called *statements.*

Every statement $\Gamma_i$ is a primitive operation $\omega_{j_i}$ of input size $s_{j_i}$ and $s_{j_i}$ *pointers* to the input variables $u_1^{(i)}, \ldots, u_{j_i}^{(i)} \in [-n, \ldots, i-2] \in \mathbb{Z}$. The *length* of the GSP is $m$.

The notion of *geometric straight-line programs* is the basis of the way geometry is modelled in Cinderella. For a brightening example, see [9, p. 43].

## 2.2 Projective Dynamic Geometry

Chapter 5.1 and 5.2 in [9] describes the way points and lines are internally represented in Cinderella. This is done by means of *homogeneous coordinates.*

The *points* are 1-dimensional linear subspaces of $\mathbb{R}^3$, the *lines* are 2-dimensional linear subspaces of $\mathbb{R}^3$. Since every 1-dimensional subspace $U \subset V$ can be written as

$$U = \{\lambda x, \lambda \in \mathbb{R}\}$$

we can identify points in the real projective plane with antipodal point-pairs on the unit sphere $S_2$. For lines we can work with the orthogonal complement of the subspace, which is of course 1-dimensional, and we can identify antipodal point-pairs with these complements.

Points are represented by three coordinates $(x : y : z)$, not all being zero. For example, $p := (x : y : z)$ defines a unique subspace of $\mathbb{R}^3$ that contains the origin and $p$. Since all $\lambda(x : y : z)$ represent the same subspace $(\lambda \neq 0)$, and because of that the same point, we identify scalar multiples, i.e. $\lambda(x : y : z)$ is identified with $(x : y : z)$ for all $\lambda \neq 0$.

Demanding that at least one of the three coordinates is not equal to zero, means the points used are in an affine plane in $\mathbb{R}^3$.

On first sight this transformation only complicates things: Why not use the representation $(x, y)$ we are all used to? However, this particular way of representing points and lines has much advantages in dynamic geometry. For example, to test if a point is on a line, no matter which representation of the elements, the scalar product of a line and a point must be zero. This is the case since the two 1-dimensional subspaces must have a non-trivial intersection. Using the same reasoning we can obtain similar expressions for perpendicularity, collinearity and concurrency.

The interested reader is encouraged to read Chapter 5 in [9] for a more detailed description.

## 2.3 Randomized Proving

In [9, Chapter 5.3] a way is described to prove geometric theorems automatically without using symbolic methods like Gröbner Bases. Although this report focuses on symbolic methods, the (typically slow) symbolic prover will be used on theorems that the (typically fast) randomized prover has determined to be true.

For randomized theorem proving we need the following lemma:

**Lemma 2.3.1.** ***(Test Set Lemma)*** *Let $Q(x_1, \ldots, x_n) \in \mathbb{K}[x_1, \ldots, x_n]$ be a multivariate polynomial of degree in $x_i$ less than or equal to $d_i$. Fix finite subsets $S_i \subset \mathbb{K}$ with $|S_i| > d_i$. If $Q(r_1, \ldots, r_n) =$ for all $(r_1, \ldots r_n) \in S_1 \times \ldots \times S_n$ then $Q \equiv 0$.*

We can work with theorems by the following two definitions:

**Definition 2.3.2. (Constructive Incidence Statement)** A *Constructive Incidence Statement* is a GSP $(X, R, \Gamma)$ with input size $n$ and length $m$ on a homogeneous RIS with the following properties:

1. All input variables are of type $O_1$, i.e. they stand for homogeneous vectors.
2. All intermediate results are of type $O_1$, except for $R_{m-1}$, which is of type $O_2$. The variable $R_{m-1}$ is called *conclusion*.

**Definition 2.3.3. (Truth of a Constructive Incidence Statement)** A *Constructive Incidence Statement* of length $m$ is true if and only if all instances of it have $R_{m-1} = 0$, i.e. the polynomial encoded by $R_{m-1}$ is the zero polynomial.

With Lemma 2.3.1 we are able to find out whether a Constructive Incidence Statement is true or not. This is what Cinderella does automatically, although some optimizations are used to reduce the number of points to be tested.

# Chapter 3

# Algebraic Proofs

In this chapter it is tried to present some knowledge, skills and tricks that could be of aid with the eventual implementation of geometric theorem proving using algebraic methods. Most of the information on the Gröbner basis algorithm is taken from [10], the parts on homogeneous ideals are mainly from [1]. This information should help us to make an implementation in such a way that the Gröbner basis algorithm is used as efficiently as possible.

## 3.1 What is a Module?

Later on in this report (when obtaining algebraic proofs) we use modules and ideals. Modules are sets with the following properties [10, p. 18]:

**Definition 3.1.1. (Module)** An *$R$-module* $M$ of a ring $R$ is a commutative group $(M, +)$ with an operation $\cdot : R \times M \mapsto M$ (called *scalar multiplication*) such that $\forall m \in M : 1 \cdot m = m$, and the associative and distribution laws are satisfied. A commutative subgroup $N \subseteq M$ is called an *$R$-submodule* if we have $R \cdot N \subseteq N$. If $N \subset M$ then it is called a *proper* submodule. An $R$-submodule of the $R$-module $R$ is called an *ideal* of $R$.

**Definition 3.1.2.** A module can have one or more of the following properties [10, p. 19]:

1. A set $\{m_1, \ldots, m_r\}$ of elements of $M$ is called a *system of generators* of $M$ if every $m \in M$ has a representation $m = f_1 m_{\lambda_1} + \ldots + f_n m_{\lambda_n}$ such that $n \in \mathbb{N}$ and $f_i \in R$. In this case we write $M = \langle m_1, \ldots, m_r \rangle$.

2. The module $M$ is called *finitely generated* if it has a finite system of generators. If $M$ is generated by a single element, it is called *cyclic*. A cyclic ideal is called a *principal ideal*.

3. A system of generators $\{m_1, \ldots, m_r\}$ is called an *$R$-basis* if every element $m \in M$ has a unique representation as above. If $M$ has an $R$-basis, it is called a *free $R$-module*.

4. If $M$ is a finitely generated free $R$-module and $\{m_1, \ldots, m_r\}$ is an $R$-basis of $M$, then $r$ is called the *rank* of $M$ and denoted by $\mathrm{rk}(M)$. The rank of $M$ is well-defined, since all bases of a finitely generated free module have the same length.

If the module $M$ is generated by the set $G = \{g_1, \ldots, g_n\}$ we will write $M = (G)$ or $M = (g_1, \ldots, g_n)$.

## 3.2 What is a Gröbner basis?

The text below was inspired by [10, Chapter 0.2], although it can be found in any book on commutative algebra.

Let

$$c_1(x_1, \ldots, x_n) = 0, \ldots, c_k(x_1, \ldots, x_n) = 0$$

be a system of polynomial equations over a certain field, and let $t(x_1, \ldots, x_n) = 0$ be an additional polynomial equations. How can we decide if $t(x_1, \ldots, x_n) = 0$ holds for all solutions of the original system of polynomial equations? The problem can be solved by checking if $t$ is in the ideal $I = (c_1, \ldots, c_k)$, because if $t \in I$, then

$$t = f_1 c_1 + \ldots + f_k c_k \tag{3.1}$$

and thus for every solution of the system of polynomial equations $t = 0$.

The problem of checking if $t \in I$ is called the *Ideal Membership Problem*. This is where Gröbner bases are used: A Gröbner basis is a special system of generators of the ideal $I$ with the property that the decision whether $t \in I$ or not, can be answered by a simple division with remainder process. From [10, p. 111] we pick the following characterization of Gröbner basis.

**Theorem 3.2.1.** ***(Characterization of Gröbner basis)*** *For a set of elements $G = \{g_1, \ldots, g_k\} \subseteq P^r \backslash \{0\}$ which generates a submodule $M = (g_1, \ldots, g_k)$, the following conditions are equivalent:*

1. *$G$ is a Gröbner basis of $M$,*

2. *For every element $m \in M \backslash \{0\}$, there are $f_1, \ldots, f_k \in P$ such that $m = \sum_{i=1}^{k} f_i g_i$,*

3. *For an element $m \in P^r$ we have that the remainder on division of $m$ by $G$ is equal to zero if and only if $m \in M$,*

4. *The set $\{\mathrm{lt}(g_1), \ldots, \mathrm{lt}(g_k)\}$ generates the $P$-submodule $\mathrm{lt}(M)$ of $P^r$, where $\mathrm{lt}(f)$ denotes the leading term of $f$.*

There is an explicit algorithm which allows us to find a Gröbner basis of an ideal $I$, starting with any set of generators of $I$. This algorithm is called Buchberger's Algorithm [10, p. 123, 125].

## 3.3 Geometric Theorems and Ideals

**Definition 3.3.1. (Geometric Theorem)** We will express a geometric theorem as follows, in the ring $R = K[x_1, \ldots, x_n]$:

- The configuration is expressed as polynomial equations $c_1 = 0, \ldots, c_n = 0$, where $c_i \in R$,
- The thesis is expressed as polynomial equation $t = 0$, where $t \in R$.

**(Truth of a Geometric Theorem)** We will say our thesis *holds* if:

$$\forall (x_1, \ldots, x_n) \in K^n : c_1(x_1, \ldots, x_n) = \ldots = c_n(x_1, \ldots, x_n) = 0 : t(x_1, \ldots, x_n) = 0. \tag{3.2}$$

In words: All instances that meet the polynomial equations describing the configuration, should meet the polynomial equation describing the thesis. It appears this can be checked using ideals, as is explained in [10, Section 2.6].

**Definition 3.3.2. (Zeroset)** Let $K \subset L$ be a field extension, let $\overline{K}$ be the algebraic closure of $K$, and let $P = K[x_1, \ldots, x_n]$.

a) An element $(a_1, \ldots, a_n) \in L^n$ (a *point* of $L^n$) is said to be a *zero* of a polynomial $f \in P$ in $L^n$ if $f(a_1, \ldots, a_n) = 0$. The set of all zeros of $f$ in $L^n$ will be denoted by $\mathcal{Z}_L(f)$.

b) For an ideal $I \subseteq P$, the *set of zeros* or *zeroset* of $I$ in $L^n$ is defined as

$$\mathcal{Z}_L(I) = \{(a_1, \ldots, a_n) \in L^n | \forall f \in I : f(a_1, \ldots, a_n) = 0\}.$$

The set of zeros of $I$ in $\overline{K}^n$ will be denoted by $\mathcal{Z}(I)$.

Using sets of zeros of ideals we can find the exact requirements needed for a geometric theorem to hold in the sense of Definition 3.3.1.

**Theorem 3.3.3.** ***(Weak Nullstellensatz)*** *Let $K$ be a field, and let $I$ be a proper ideal of $P = K[x_1, \ldots, x_n]$. Then $\mathcal{Z}(I) \neq \emptyset$.*

**Proof** Let $\overline{K}$ be the algebraic closure of $K$, and let $\overline{P} = \overline{K}[x_1, \ldots, x_n]$. Then $I\overline{P}$ is a proper ideal of $\overline{P}$. Furthermore, $I\overline{P}$ is contained in some maximal ideal $J$ of $\overline{P}$, because $\overline{P}$ is Noetherian. By [10, Corollary 2.6.9] there exists a point $(a_1, \ldots, a_n) \in \overline{K}^n$ such that $J = (x_1 - a_1, \ldots, x_n - a_n)$. Hence, $(a_1, \ldots, a_n)$ is a zero of $J$, so $(a_1, \ldots, a_n)$ is a zero of $I \subseteq I\overline{P} \subseteq J$. □

**Corollary 3.3.4.** *Let $L$ be a field which contains the algebraic closure of $K$, and let $I$ be an ideal of $K[x_1, \ldots, x_n]$. Then we have the following equivalence relation:*

$$\mathcal{Z}_L(I) = \emptyset \Leftrightarrow 1 \in I.$$

**Proof** The "$\Leftarrow$" is clear. For the "$\Rightarrow$" we observe that $\mathcal{Z}_L(I) = \emptyset$ implies $\mathcal{Z}(I) = \emptyset$, so (by the Weak Nullstellensatz) $I$ is not a proper ideal, so $1 \in I$. □

**Definition 3.3.5. (Radical Ideal)** Let $R$ be a ring, and $I$ an ideal in $R$. The set

$$\sqrt{I} := \{r \in R | r^i \in I \text{ for some } i \geq 0\}$$

is again an ideal of $R$. This ideal is called the *radical* of $I$. An ideal $I$ such that $I = \sqrt{I}$ is called a *radical ideal.*

If $I$ is an ideal of $K[x_1, \ldots, x_n]$ it is clear that $I$ and $\sqrt{I}$ have the same set of zeros.

**Definition 3.3.6. (Vanishing Ideal)** Let $K \subseteq L$ be a field extension, let $S \subseteq L^n$. Then the set of all polynomials $f \in K[x_1, \ldots, x_n]$ such that $f(a_1, \ldots, a_n) = 0$ for all points $(a_1, \ldots, a_n) \in S$ forms an ideal of the polynomial ring $K[x_1, \ldots, x_n]$. This ideal is called the *vanishing ideal* of $S$ and is denoted by $\mathcal{I}(S)$.

**Theorem 3.3.7.** ***(Hilbert's Nullstellensatz)*** *Let $K$ be an algebraically closed field, and let $I$ be a proper ideal of $K[x_1, \ldots, x_n]$. Then*

$$\mathcal{I}(\mathcal{Z}(I)) = \sqrt{I}.$$

**Proof** The proof consists of two parts.

- "$\mathcal{I}(\mathcal{Z}(I)) \supseteq \sqrt{I}$": Suppose $f \in \sqrt{I}$, so $f^t \in I$ for some $t \in \mathbb{N}$. By definition, for every $\underline{a} \in \mathcal{Z}(I)$ we have $f^t(\underline{a}) = 0$, so we have $f(\underline{a}) = 0$ for all $\underline{a} \in \mathcal{Z}(I)$, so $f \in \mathcal{I}(\mathcal{Z}(I))$.

- "$\mathcal{I}(\mathcal{Z}(I)) \subseteq \sqrt{I}$": Assume $I \neq (0)$, and define the ring $P = K[x_1, \ldots, x_n]$. Choose $f \in \mathcal{I}(\mathcal{Z}(I)) \backslash \{0\}$, and a system of generators $\{g_1, \ldots, g_k\}$ of $I$. Let $y$ be a new indeterminate and consider the ideal $I' = IP[y] + (yf - 1)$ in the ring $P[y]$. Now suppose the point $(a_1, \ldots, a_n, b)$ is in $\mathcal{Z}(I')$, then we have $bf(a_1, \ldots, a_n) = 1$ and $g_i(a_1, \ldots, a_n) = 0$ for $i = 1, \ldots, k$. But then $(a_1, \ldots, a_n) \in \mathcal{Z}(I)$ and $f(a_1, \ldots, a_n) \neq 0$, which is a contradiction with the choice of $f$. This means we cannot choose such a point in $\mathcal{Z}(I')$, which means $\mathcal{Z}(I') = \emptyset$. By the Weak Nullstellensatz we know $1 \in I'$, and by theorem 3.3.8 we now have $f \in \sqrt{I}$.

□

Now suppose we have a configuration as described in Definition 3.3.1. Observe the ideal $I = \mathbb{Q}(c_1, \ldots, c_k)$. We write $\underline{X}$ for $(x_1, \ldots, x_n)$. By Definitions 3.3.2, 3.3.5 and 3.3.6 and Theorem 3.3.7 we know that the following expressions are equivalent:

- $t \in \sqrt{I}$,
- $t \in \mathcal{I}(\mathcal{Z}(I))$,
- $t(\underline{X}) = 0$ for all points $\underline{X} \in \mathcal{Z}(I)$,
- $t(\underline{X}) = 0$ for all points $\underline{X}$ for which $c_1(\underline{X}) = \ldots = c_n(\underline{X}) = 0$,
- The theorem holds in the sense of Definition 3.3.1.

So the theorem can be proven if we can decide whether $t \in \sqrt{I}$. We have the following theorem to help us make that decision:

**Theorem 3.3.8.** *Let $I$ be an ideal in $K[x_1, \dots, x_n]$, and let $f \in K[x_1, \dots, x_n]$. Then the following equivalence holds:*

$$f \in \sqrt{I} \Leftrightarrow 1 \in (f_1, \dots, f_k, zf - 1) \tag{3.3}$$

*where $(f_1, \dots, f_k, zf - 1)$ is an ideal in $K[x_1, \dots, x_n, z]$ and $z$ is a new indeterminate.*

**Proof** This proof was taken from [19] and adapted.

- "$\Rightarrow$". Let $f \in \sqrt{I}$, so $f^m \in I$ for some $m \in \mathbb{N}$. So

  $$z^m f^m \in I \subseteq (f_1, \dots, f_k, zf - 1) \subseteq K[x_1, \dots, x_n, z],$$

  and

  $$1 - f^m z^m = (1 - fz)(1 + fz + \dots + (fz)^{m-1}) \in (zf - 1) \subseteq (f_1, \dots, f_k, zf - 1),$$

  so

  $$1 = \underbrace{1 - f^m z^m}_{\in (f_1, \dots, f_k, zf-1)} + \underbrace{f^m z^m}_{\in (f_1, \dots, f_k, zf-1)} \in (f_1, \dots, f_k, zf - 1).$$

- "$\Leftarrow$" Let $1 \in (f_1, \dots, f_k, zf - 1)$. Then:

  $$1 = \alpha_1 f_1 + \dots + \alpha_k f_k + \alpha(zf - 1), \text{ where } \alpha_i, \alpha \in K[x_1, \dots, x_n, z].$$

  We proceed to $K[x_1, \dots, x_n, z]$ and substitute: $z := \frac{1}{f}$. We obtain:

  $$1 = \alpha'_1 f_1 + \dots + \alpha'_k f_k$$

  Both sides are multiplied by $f^t$, where $t$ is the maximum power of $z$ that occurs in $\alpha_i, \alpha$. We obtain

  $$f^t = \beta_1 f_1 + \dots + \beta_k f_k$$

  where $\beta_i \in K[x_1, \dots x_n]$, so $f^t \in I$ and $f \in \sqrt{I}$.

□

The second part of the proof is illustrated by the following example.

**Example** Observe the ideal $I = (x^2 + y, y^2) \subset \mathbb{Q}[x, y]$. We want to know if the polynomial $f(x, y) = x$ is in $\sqrt{I}$. Following Theorem 3.3.8 we proceed to the ideal $J = (x^2 + y, y^2, fz - 1)$ in $\mathbb{Q}[x, y, z]$. We find $1 \in J$, because

$$1 = (z^2 - yz^4) \cdot (x^2 + y) + z^4 \cdot y^2 + (xyz^3 + yz^2 - xz - 1) \cdot (fz - 1).$$

As described in the proof above, we substitute $z := \frac{1}{f}$ and multiply both sides by $f^4$, i.e. replace $z^i$ by $f^{4-i}$. Observe how the last term (always!) falls out and we obtain

$$f^4 = (f^2 - y) \cdot (x^2 + y) + 1 \cdot y^2,$$

which shows indeed $f \in \sqrt{I}$ and even gives a proof of that assertion.

Now the problem remains how to find out if $t \in \sqrt{I}$. As stated in [10, p. 114]:

**Lemma 3.3.9.** ***(Ideal Membership)*** *Let $I$ be an ideal in $K[x_1, \ldots, x_n]$, $B$ a Gröbner basis of $I$ and $f \in K[x_1, \ldots, x_n]$. Then the following holds:*

$$f \in I \text{ if and only if the remainder on divisien of } f \text{ by } B \text{ is } 0.$$

**Remark 3.3.10.** In practice, testing if $f \in I = (c_1, \ldots, c_k)$ often suffices. That is why we will focus on that particular demand for now, especially while the advantages of homogeneity (as described in Section 3.6) are lost when proceeding to the ideal $(c_1, \ldots, c_k, zf - 1)$.

Summarizing this section, we have the following lemma to test if a geometric theorem holds:

**Lemma 3.3.11.** ***(Testing if a Geometric Theorem holds)*** *A geometric theorem given by configuration polynomials $c_1, \ldots, c_k$ and thesis polynomial $t$ holds if the remainder on division of $t$ by $B$ is equal to zero, where $B$ is a Gröbner basis of the ideal generated by $(c_1, \ldots, c_k)$. Moreover, that theorem holds if* and only if *the remainder on division of $1$ by $B'$ is equal to zero, where $B'$ is a Gröbner basis of the ideal generated by $(B, zt - 1)$, where $z$ is a new indeterminate.*

## 3.4 Obtaining a Certificate by Modules

We would like to be able to find the $f_i$ from Definition 3.3.1, as these can serve as some kind of certificate to prove the correctness of what we have calculated. The verification of $t = f_1c_1 + \ldots + f_nc_n$ is straightforward and does not require any knowledge on the procedure used to obtain the proof. One of the advantages of this 'certificate' is that any calculation or algorithm can be used, as long as correct $f_i$ are returned. This enables us to use tricks we cannot prove to be mathematically correct, or make assumptions that we cannot prove to be satisfied. As long as we return a correct set $(f_1, \ldots, f_n)$ we can do whatever we like.

Throughout this section we assume the thesis holds, so

$$\exists f_1, \ldots, f_n : t = f_1c_1 + \ldots + f_nc_n.$$

One way to obtain these $f_i$ is by means of modules.

**Algorithm 3.4.1.** Again, the configuration is given by polynomial equations $c_1 = \ldots = c_n = 0$ and the thesis is given by the polynomial equation $t = 0$.

1. Define the polynomial vector $d_i \in R^{n+1}$ by $d_i = c_ie_1 - e_{i+1}$, where $i = 1, \ldots, n$,

2. Define the module $M = (d_1, \ldots, d_n) \subset R^{n+1}$,

3. Calculate a Gröbner basis $B$ of $M$,

4. Calculate the remainder $r \in R^{n+1}$ on division of $te_1$ by $B$, where the ordering should be such that $e_i > e_{i+1}$,

5. Since $t = f_1c_1 + \ldots + f_nc_n$ we have $r = (0, h_1, \ldots, h_n)$ for certain $h_1, \ldots, h_n \in R$.

**Theorem 3.4.2.**

$$t = h_1c_1 + \ldots + h_nc_n$$

**Proof** Let $i \in \{1, \ldots, n\}$. Then $d_i = c_ie_1 - e_{i+1} \in M$, so $e_{i+1} = c_ie_1 \mod M$.

Now $te_1 \equiv r \bmod M$, so $te_1 \equiv (0, h_1, \ldots, h_n) \equiv h_1e_2 + \ldots + h_ne_{n+1} \equiv h_1c_1e_1 + \ldots h_nc_ne_1$, so $t = h_1 + \ldots + h_n$. □

So by means of modules we can obtain the certificate. The major drawback of this method is that it tends to be very slow when adding equations, as $M \subset R^{n+1}$.

## 3.5 The Extended Buchberger Algorithm

It appears to be possible to obtain a certificate without the need to use modules. Again, we assume that the thesis holds.

**Algorithm 3.5.1.** Again, the configuration is given by polynomial equations $c_1 = \ldots = c_n = 0$ and the thesis is given by the polynomial equation $t = 0$.

1. Define the ideal $I = (c_1, \ldots, c_k)$,
2. Using the extended Buchberger Algorithm [10, p. 125], find a Gröbner basis $G = (g_1, \ldots, g_t)$ and a $n \times t$-matrix $\mathcal{A} = (a_{ij})$ such that
$$g_j = a_{1j}c_1 + \ldots + a_{nj}c_n \text{ for } j = 1, \ldots, t. \qquad (3.4)$$
3. Divide $t$ by $G$ and obtain remainder 0 (by the assumption that the thesis holds) and $h_1, \ldots, h_t \in R$ such that $t = h_1g_1 + \ldots + h_tg_t$,
4. Define $f_i = h_1a_{i1} + \ldots + h_ta_{it}$, for $i = 1, \ldots, n$.

**Theorem 3.5.2.** *At the end of this algorithm the following relation between t and the $f_i$, as defined above, holds:*

$$t = f_1c_1 + \ldots + f_nc_n,$$

which means these $f_i$ are the certificate required.

**Proof**

$$\begin{array}{rcl} t & = & h_1g_1 + \ldots + h_tg_t \\ & = & \text{(use Equation 3.4)} \\ & = & h_1(a_{11}c_1 + \ldots + a_{n1}c_n) \\ & + & \ldots \\ & + & h_t(a_{1t}c_1 + \ldots + a_{nt}c_n \\ & = & f_1c_1 + \ldots + f_nc_n. \end{array}$$

□

So the certificate can be obtained by Algorithm 3.5.1 without the use of modules.

There is another way to obtain a certificate: By use of the `division` command in SINGULAR. The documentation does not make clear how this function works, but it does make it possible to obtain the certificate without any additional programming. One should expect it to use the Extended Buchberger algorithm, since that algorithm does not really slow the process down.

When testing, it appears that this is not the case. There are examples where the calculation of the Gröbner basis of the ideal $I = (c_1, \dots, c_k)$ and $t \in I$ is done within a second, but the division of $t$ by $c_1, \dots, c_k$ takes much longer.

## 3.6 Homogeneous Ideals

In order to fully understand the notion of homogeneous polynomials and ideals we define them rather precisely, unfortunately that might make this section a bit harder to read. Most of the text below is taken from [1, p. 466-476] and [5, p. 252-254], although adjusted on occasion. For the remainder of this section, $K$ will be a field, the ring $R$ is equal to $K[x_1, \dots, x_n]$ and $T$ is the set of terms in the variables $x_1, \dots x_k$.

**Definition 3.6.1. (Grading)** A *grading* $\Gamma$ of $K[x_1, \dots, x_n]$ is a monoid homomorphism

$$\Gamma : (T, 1, \cdot) \mapsto (\mathbb{N}, (0), +),$$

i.e. a map $\Gamma : T \mapsto \mathbb{N}$ such that $\Gamma(1) = 0$ and $\Gamma(s \cdot t) = \Gamma(s) + \Gamma(t)$. For $f \in R$, $f \neq 0$, the $\Gamma$**-degree** of $f$ is defined as

$$\max\{\Gamma(t) | t \in T(f)\}.$$

By abuse of notation, the $\Gamma$-degree of $f$ is denoted by $\Gamma(f)$. By $K[x_1, \dots, x_n]_{[d_1,d_2]}$ we denote the set of polynomials with the property $d_1 \leq \Gamma(f) \leq d_2$.

**Definition 3.6.2. (Homogeneous Polynomial)** A non-zero polynomial $f \in R$ is called $\Gamma$-*homogeneous* if $\Gamma(s) = \Gamma(t)$ for all terms $s, t \in T(f)$.

**Example** Let $\alpha_1, \dots, \alpha_n \in \mathbb{N}$ and define $\Gamma : T \mapsto \mathbb{N}$ by

$$\Gamma(x_1^{a_1} \cdot \dots \cdot x_k^{a_k}) = a_1\alpha_1 + \dots + a_k\alpha_k.$$

Taking $\alpha_1 = \dots = \alpha_k = 1$ yields the *grading by total degree*, so $\Gamma(f)$ is really $\deg(f)$.

It appears we can speed up the Gröbner basis calculation when working with homogeneous ideals. We define the following modification of the standard Gröbner basis algorithm (called *GRÖBNER* in this context):

**Definition 3.6.3. ($[d_1, d_2]$-GRÖBNER)** The algorithm $[d_1, d_2]$-*GRÖBNER* is the algorithm GRÖBNER with the sole modification that it considers only those critical pairs $\{g_1, g_2\}$ that satisfy

$$d_1 \leq \Gamma(\text{lcm}(\text{lt}(g_1), \text{lt}(g_2))) \leq d_2.$$

A $d$-Gröbner basis ($d \in \mathbb{N}$) is defined to be a finite subset of $K[x_1, \dots, x_n]$ consisting of homogeneous polynomials that satisfies the equivalent conditions as stated in [1, p. 472], the most important of which is

"The remainder on division of $f$ by $G$ is zero for all $f \in (G) \cap K[x_1, \dots, x_n]_{[0,d]}$".

We will apply this algorithm in the following setting. If $f \in K[x_1, \dots, x_n]$ and $d \in \mathbb{N}$ then we denote by $f_{(d)}$ the sum of all monomials of $f$ whose $\Gamma$-degree equals $d$. It is clear that either $f_{(d)} = 0$ or $f_{(d)}$ is homogeneous with $\Gamma(f_{(d)}) = d$. In the latter case, $f_{(d)}$ is called the $d$-**homogeneous part** of $f$.

**Definition 3.6.4. (Homogeneous Ideal)** An ideal $I$ of $K[x_1, \dots, x_n]$ is called homogeneous if $f_{(d)} \in I$ for all $f \in I$ and $d \in \mathbb{N}$.

We have the following properties of homogeneous ideals:

**Theorem 3.6.5.**

1. *Suppose $F \subseteq K[x_1, \dots, x_n]$ and all $f \in F$ are homogeneous polynomials. Then the ideal $I$ generated by $F$ is homogeneous,*
2. *Every homogeneous ideal $I \subseteq K[x_1, \dots, x_n]$ has a finite basis consisting of homogeneous polynomials.*

**Proof**

1. Let the polynomial $g \in I$. Then $g = \sum_{i=1}^{r} m_i f_i$, where $f_i \in F$ and $m_i$ a monomial in $K[x_1, \dots, x_n]$. then for $d \in \mathbb{N}$:

$$g_{(d)} = \sum_{i=1}^{r} \{m_i f_i | 1 \le i \le r, \Gamma(m_i f_i) = d\},$$

so $g_{(d)} \in I$, and by Definition 3.6.4 $I$ is homogeneous.

2. Suppose $I \subseteq K[x_1, \dots, x_n]$ is an ideal. We already know that there exists a finite set $P$ of polynomials (not necessarily homogeneous) in $K[x_1, \dots, x_n]$ such that $I = (P)$. Let $F = \{p_{(d)} | p \in P, d \in \mathbb{N}\}$. Then $F$ is finite, every $f \in F$ is of course a homogeneous polynomial, and $F \subseteq I$.

   Moreover: Let $h \in I$. Then, because $P$ is a basis for $I$, the polynomial $h = \sum h_i p_i$, for some $h_i \in K[x_1, \dots, x_n]$, so $h = \sum h_i (\sum j_i f_i)$ for some $j_i$, so $F$ is a basis of $I$.

□

A $d$-Gröbner basis of an ideal $I$ is of course a $d$-Gröbner basis $G$ such that the ideal $I$ is generated by $G$. If $I$ is a homogeneous ideal and $d \in \mathbb{N}$, then $I$ has a finite basis $F$ of homogeneous polynomials and, and $[0, d]$-GRÖBNER($F$) is then a $d$-Gröbner basis of $I$.

It is even possible to make a connection between $d$-Gröbner bases and standard representations, but we will not need to go that far. The `degBound` directive in SINGULAR implements a form of the $[0, d]$-GRÖBNER-algorithm. It makes the whole algorithm considerably faster, in a certain test case the calculation took tenths of a second instead of more than half an hour.

# Chapter 4

# Translating Polynomials

There are, of course, numerous ways to translate a given configuration in the plane into polynomials. There is however a certain structure we wish to use. This makes the process clear and the intermediate results usable in other situations. This is made possible by OpenMath.

## 4.1 The OpenMath Standard

The OpenMath standard is intended for representing mathematics in such a way that mathematical objects can easily be exchanged between computer programs.

> OpenMath is an emerging standard for representing mathematical objects with their semantics, allowing them to be exchanged between computer programs, stored in databases, or published on the worldwide web. While the original designers were mainly developers of computer algebra systems, it is now attracting interest from other areas of scientific computation and from many publishers of electronic documents with a significant mathematical content. [12, Overview]

A rough overview of the standard can be found in Figure 4.1. The 3 layers are explained as follows:

[Language] The OpenMath language defines the 'grammar'. It defines notions like Variables, Constants, Errors, and Functions.

[Content Dictonaries] A Content Dictionary (CD) is (or can be) defined for each area of Mathematics. For example the 'arith1' CD describes the notions of 'minus', 'plus', 'power', etc.

[Phrasebooks] A Phrasebook provides communication between OpenMath and another program. Phrasebooks exist for, for example, Mathematica, GAP and SINGULAR. A specific Phrasebook consists of three parts:

- An *encoder* to encode OpenMath objects into commands that the program understands,
- A *decoder* to translate program output into OpenMath objects,

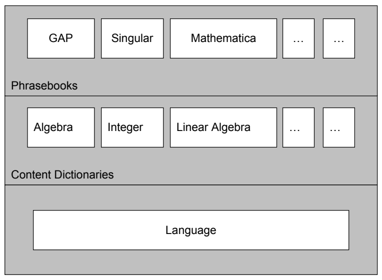

Figure 4.1: The OpenMath framework

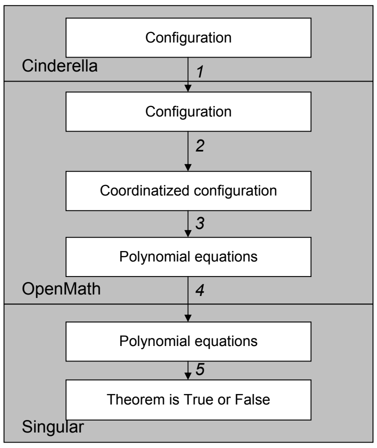

Figure 4.2: The chain of translations

- The physical communication between the program and the Java (or C, or C++) program containing the OpenMath objects.

The interested reader is encouraged to have a look at `http://www.openmath.org` for an extensive overview of the OpenMath standard.

## 4.2 The plangeo CoDec

In the (experimental) `plangeo` codec [13] a way is proposed to encode a configuration in the plane into polynomials. Moreover, the `extended_in` in the `polygb2` codec [14] enables us to represent the set of polynomials in a convenient way, before translating them to SINGULAR. This is also the OpenMath object that was used in my Bachelor's project [18].

The initial layout from Figure 1.2 was slightly changed to conform to the `plangeo`-codec, see Figure 4.2. Thus, we create the following chain of translations:

1. **Cinderella**: A configuration described by points and lines, some objects may have coordinates,
2. **OpenMath**: A configuration described by points and lines, as in `plangeo`, some objects may have coordinates,
3. **OpenMath**: A configuration described by points and lines, as in `plangeo`, all objects have coordinates,
4. **OpenMath**: A set of polynomials describing the ideal and the polynomials to be tested,
5. SINGULAR: A set of commands describing the polynomials, the ideal and the polynomials to be tested,
6. SINGULAR: True or false, including a proof if true.

The following OpenMath objects are used:

**Step** 2 An OpenMath Application with head symbol `plangeo1.assertion`,

**Step** 3 An OpenMath Binding with head symbol `fns1.lambda` and as bound variables the coordinates that had to be added to make the configuration coordinatized. The body of the Binding is the `plangeo1.assertion` from the previous step, where an Application with head `plangeo4.set_coordinates` is added to the points and lines that did not have coordinates,

**Step** 4 An OpenMath Application with head symbol `polygb2.extendend_in`, and as variables the bound variables from the Binding in the previous step. The configuration polynomials are generated from the `plangeo1.configuration` that is contained in the `plangeo1.assertion`, as is the thesis polynomial.

This repeatedly translating of the configuration may seem a bit overdone, in practice it helps us to make an implementation that is robust and easy to understand. Moreover, the intermediate results can be used by other applications, and other applications can use the single steps we created.

# Chapter 5

# Gröbner Bases in Practice

## 5.1 The Triangle

This chapter started out as a demo of how things should go, but it ended up as a demo of how things can go wrong. Suppose we want to prove that the three medians of a triangle go through one point. In this section, we will not work with homogeneous coordinates yet, to make the formulas look more familiar. The configuration is created as follows (Figure 5.1):

1. Choose points $A$, $B$, and $C$,
2. $D$ is half-way between $A$ and $B$ $(c_1, c_2)$,
3. $E$ is half-way between $B$ and $C$ $(c_3, c_4)$,
4. $F$ is half-way between $A$ and $C$ $(c_5, c_6)$,
5. $d$ is the line through $B$ and $F$ $(c_7, c_8)$,
6. $e$ is the line through $C$ and $D$ $(c_9, c_{10})$,
7. $f$ is the line through $A$ and $E$ $(c_{11}, c_{12})$,
8. $G$ is the intersection of $d$ and $e$ $(c_{13}, c_{14})$.

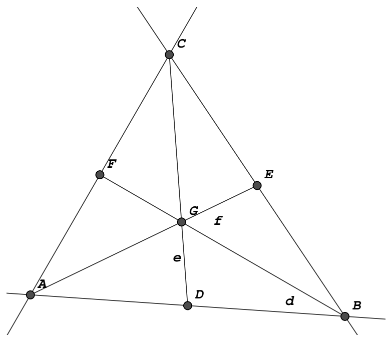

Figure 5.1: The medians of a triangle

| | |
|---|---|
| $c_1 = pA_1 + pB_1 - 2 \cdot pD_1$ | $c_9 = le_1 \cdot pC_1 + le_2 - pC2$ |
| $c_2 = pA_2 + pB_2 - 2 \cdot pD_2$ | $c_{10} = le_1 \cdot pD_1 + le_2 - pD_2$ |
| $c_3 = pC_1 + pB_1 - 2 \cdot pE_1$ | $c_{11} = lf_1 \cdot pA_1 + lf_2 - pA_2$ |
| $c_4 = pC_2 + pB_2 - 2 \cdot pE_2$ | $c_{12} = lf_1 \cdot pE_1 + lf_2 - pE_2$ |
| $c_5 = pC_1 + pA_1 - 2 \cdot pF_1$ | $c_{13} = ld_1 \cdot pG_1 + ld_2 - pG_2$ |
| $c_6 = pC_2 + pA_2 - 2 \cdot pF_2$ | $c_{14} = le_1 \cdot pG_1 + le_2 - pG_2$ |
| $c_7 = ld_1 \cdot pB_1 + ld_2 - pB_2$ | |
| $c_8 = ld_1 \cdot pF_1 + ld_2 - pF_2$ | $t = lf_1 \cdot pG_1 + lf_2 - pG_2$ |

Table 5.1: The system of polynomials

Our hypothesis $t$ is, of course, that $G$ lies on $f$. The translation into polynomials can be found in Table 5.1. Variables starting with a 'p' denote points, variables starting with a 'l' denote lines.

It appears that $t$ is not in the ideal generated by $c_1, \ldots, c_{14}$. Obviously, this does not mean the theorem is false. It does mean that we failed to translate the theorem into polynomials in such a way that the truth of the theorem can be proved. In practice, this kind of behaviour occurs because of degenerations. For example, a line through the $A$ and $B$ is not correctly defined if $A = B$. A common way to fix this is adding 3 equations ($c_{15}, c_{16}, c_{17}$) to tell $A \neq B \neq C \neq A$, of the form

$$c_{15} = (pA_1 - pB_1) * z_1 - 1,$$

where $z_1$ is a new ring variable. Obviously, if $pA_1 = pB_1$, the polynomial $c_{15}$ will never evaluate to zero. By doing this, we (so to say) explicitly removed the situations where $pA_1 = pB_1$ from the configuration. Sadly, this does not work in this case.

Another situation we might want to avoid is when $A$, $B$, and $C$ are on one line. This can be done by adding a line $z$ through $A$ and $B$, where $C$ should not be on (again, we add $c_{15}, c_{16}, c_{17}$):

$$\begin{aligned} c_{15} &= lz_1 \cdot pA_1 + lz_2 - pA_2, \\ c_{16} &= lz_1 \cdot pB_1 + lz_2 - pB_2, \\ c_{17} &= (lz_1 \cdot pC_1 + lz_2 - pC_2) \cdot z - 1, \end{aligned}$$

where $lz_1, lz_2$, and $z$ are new variables. The SINGULAR session that tells us whether $t \in I$ (the declaration of the polynomials is omitted):

```
> ideal i = (c1,c2,c3,c4,c5,c6,c7,c8,c9,c10,c11,c12,c13,c14,c15,c16,c17);
> reduce(t,groebner(i));
0
```

It appears that we found a proof for this theorem. Unfortunately, this is a pretty ad-hoc way of solving things, and that is not something desirable (nor achievable) in the eventual solution.

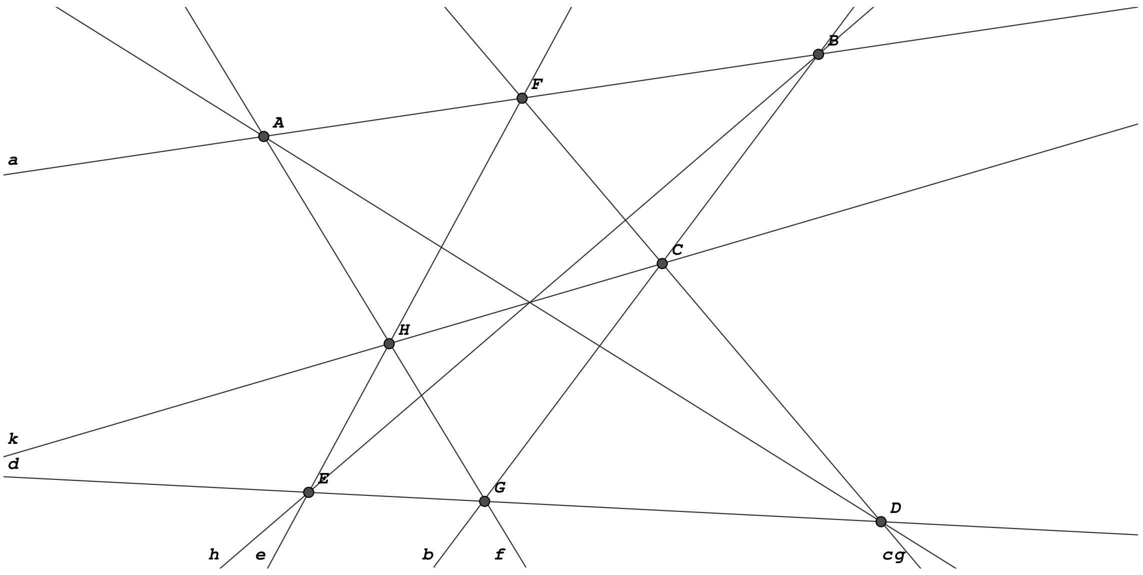

Figure 5.2: Pappos' theorem

## 5.2 Pappos' Theorem - 1

In the following sections two different methods of representing Pappos' theorem will be presented. The encoding is done in such a way that it is possible to create an algorithm that generates these polynomials from a configuration that was drawn by a user. The purpose of these sections is to make clear how a certain configuration can be encoded, and to demonstrate the problems encountered.

The configuration we will be observing is shown in Figure 5.2, which was created in Cinderella. The thesis is, of course, that lines $g$, $h$, and $k$ go through one point. We notice that this theorem only uses the `Join` and `Meet`, as defined in Section 2.1. We recall their 'loose' definitions:

- $a$ is `Join`$(A, B)$: The line $a$ is the line through the points $A$ and $B$,
- $A$ is `Join`$(a, b)$: The point $A$ is the point where the lines $a$ and $b$ meet.

The algorithm by which the configuration from Figure 5.2 was created can be found in Section A.1.

We will use the representation of points and lines in homogeneous coordinates as described in Section 2.2, as that gives us homogeneous polynomials, with all the advantages from Section 3.6. The most important result from those sections is that a point is on a line if and only if the scalar product of their coordinates is zero. In the equations that follow, a point $A$ is denoted by the three coordinates $(pA_x, pA_y, pA_z)$, a line $a$ is denoted by $(la_x, la_y, la_z)$. Thus, the algorithms called `Join` and `Meet` are encoded as follows:

`a := Join(A,B)` The point $A$ is on the line $a$, and so is the point $B$:

$$\begin{array}{rcl} pA_x \cdot la_x + pA_y \cdot la_y + pA_z \cdot la_z & = & 0, \\ pB_x \cdot la_x + pB_y \cdot la_y + pB_z \cdot la_z & = & 0. \end{array} \quad (5.1)$$

`A := Meet(a,b)` The point $A$ is on the line $a$, and on the line $b$:

$$\begin{aligned} pA_x \cdot la_x + pA_y \cdot la_y + pA_z \cdot la_z &= 0, \\ pA_x \cdot lb_x + pA_y \cdot lb_y + pA_z \cdot lb_z &= 0. \end{aligned} \tag{5.2}$$

The thesis that the lines $g$, $h$, and $k$ go through one point is encoded by adding a point $K = \texttt{Meet}(h, k)$ and describing the incidence of the point $K$ and the line $g$ in the thesis polynomial. The 26 equations describing the configuration we obtain can be found in Section B.1. The polynomial describing the thesis is:

$$t = lg_x \cdot pK_x + lg_y \cdot pK_y + lg_z \cdot pK_z. \tag{5.3}$$

At this point in time we have all the ingredients we need to test the thesis with a computer algebra package, for example SINGULAR. As the polynomials $c_1, \ldots, c_{26}$ and $t$ are homogeneous with degree 2, we can use a 2-Gröbner basis $G$ of the (homogeneous) ideal generated by $(c_1, \ldots, c_{26})$. This is done in a split second, but the remainder $r$ on division of $t$ by $G$ is not equal to zero, so we do not have a proof yet.

Looking at the equations carefully, we see that the exact thing that helps us when obtaining the Gröbner basis, the homogeneity, now stops us from proving the theorem. Suppose all variables are set to 0, except $lg_x$ and $pK_x$ are set to 1. The fact that the polynomials are homogeneous makes this a valid instance of the configuration, $c_1 = \ldots = c_{26} = 0$. However, the thesis polynomial $t$ is not equal to 0. This explains why $r \neq 0$. This problem can be solved by demanding that for every point and every line the first coordinate ($x$) should be not-zero. Remember that in Euclidian geometry this is no real restriction. Since all theorems can be moved around freely in the plane, it is always possible to move the configuration away from these critical positions. There are two ways to add these constraints to our model.

Firstly, we can try to add 18 polynomials (9 for the points, 9 for the lines) to the *configuration*. These polynomials are of the form $d_i = ??_x \cdot z_i - 1$. Obviously, when $pA_x$ is equal to zero, $d_1$ can never be equal to zero. Thus, we removed the instances where $pA_x$ is zero explicitly from the configuration. However, we now have an ideal generated by 26 + 18 = 44 polynomials, and at the same time we have lost the possibility to use all the beautiful tools we have for homogeneous ideals. The combination of these two factors makes sure the Gröbner basis calculation is too slow.

Secondly, we can try to add 18 factors to the *thesis*-polynomial, thus testing if $pA_x \cdot pB_x \cdot \ldots \cdot lk_x \cdot t$ is in the ideal generated by $c_1, \ldots, c_{26}$. If it is, we know that if the first coordinate of each variable is not equal to zero, the thesis polynomial $t$ must be equal to zero, so our thesis holds. However, the polynomial for which we want to know if it is in $(c_1, \ldots, c_{26})$ now has degree 18 + 2 = 20. This means we need a 20-Gröbner basis instead of a 2-Gröbner basis, and again, this is too much to ask for. The calculation in SINGULAR fails because it runs out of memory.

It appears this set of polynomials does not give us a proof of Pappos' theorem.

## 5.3 Pappos' Theorem - 2

As stated in Section 2.1, three points $A$, $B$, and $C$ are collinear if and only if the determinant

$$\begin{vmatrix} pA_x & pB_x & pC_x \\ pA_y & pB_y & pC_y \\ pA_z & pB_z & pC_z \end{vmatrix}$$

is zero. This can dramatically reduce the amount of polynomials needed to describe a configuration. As shown in section 8.1 it is possible to obtain a list of collinearity conditions from an algorithm such as the one in Section A.1.

In the case of Pappos, the entire configuration can be described by just 8 (homogeneous!) polynomials. Looking at Figure 5.2, we see that the following triplets of points are collinear: $ABF$, $EGD$, $EFH$, $CDF$, $AHG$, $GCB$, $AKD$, and $EKB$. The thesis is that the points $H$, $K$, and $C$ are collinear. The configuration polynomials can be found in Section B.2, the thesis polynomial is:

$$\begin{aligned} t &= pH_x \cdot (pK_y \cdot pC_z - pC_y \cdot pK_z) \\ &- pK_x \cdot (pH_y \cdot pC_z - pC_y \cdot pH_z) \\ &+ pC_x \cdot (pH_y \cdot pK_z - pK_y \cdot pH_z). \end{aligned} \tag{5.4}$$

Again, since $c_1, \ldots, c_8$ are homogeneous with degree 3, and $t$ has degree 3, we can calculate a 3-Gröbner basis $G$ of the ideal $I$ generated by $(c_1, \ldots, c_8)$. The remainder on division of $t$ by $G$ is not equal to zero. Again, there is some kind of situation that meets the configuration, but does not meet the thesis.

However, the situation is much more complicated here than in the previous section. Demanding that the first coordinate of all the variables is not equal to zero does not suffice: For example, set all the first coordinates to $1$, and second and third to $0$, except for $pK_y$ and $pC_z$. Again, all the configuration equations are met, but the thesis polynomial is not. This particular type of degeneration makes it hard to come up with the required additional polynomials, let alone the evaluation in SINGULAR, which will probably be just as slow as in the previous section.

This is a problem that occurs every time we try to prove a theorem without any optimizations made by a person. Articles exist where the authors succeeded in proving a lot of geometrical theorems with the aid of Gröbner bases, for example [2, 3, 7, 8, 11]. Unfortunately, in all these articles the geometry theorems tested were translated into polynomials *by hand*. This gives one the possibility to create a 'good' translation, for example rotate and scale the configuration in such a way that one point is the origin and another point is on the $x$-axis. One could also exploit certain symmetries in the configuration or make generalizations that simplify the resulting polynomials. All these optimizations can only be done by a person, preferably a mathematician, looking at the configuration and encoding it in an efficient way. This is something that can *not* be done by a computer program.

# Chapter 6

# Projective Geometry

As shown in the previous chapter, Gröbner Bases are simply too slow to efficiently automatically prove geometric theorems. In this chapter we restrict ourselves to geometric theorems that are invariant under projective transformations, based on a paper by Jürgen Richter-Gebert in 1995 [16]. This means we only consider configurations and theses of the form:

- The three points $A$, $B$, and $C$ lie on one line (the points $A$, $B$, and $C$ are collinear), denoted by '$h(A, B, C)$',
- The three lines through $A$ and $B$, $C$ and $D$, and $E$ and $F$, respectively, go through one point, denoted by '$m((A, B), (C, D), (E, F))$',
- The six points $A$, $B$, $C$, $D$, $E$, and $F$ lie on one conic, denoted by '$c(A, B, C, D, E, F)$'.

**Remark 6.0.1.** It suffices to observe cases where, for example, all collinearity conditions are given by three points. In the case that four or more (say $N$) points are on one line, we simply add the $\binom{N}{3}$ possible collinearity conditions on three points to the configuration.

In the mathematical foundation in this chapter parts from the masters thesis by one of Jürgen Richter-Gebert's students, Andreas Umbach, were consulted [22].

The properties above are all invariant with respect to some group of linear transformations, which enables us to use *bracket algebra* [21, Chapter 3]. We again observe the homogeneous coordinates in the plane. This means the coordinates are in $(\mathbb{R}^3\backslash\{0\})/\mathbb{R}\backslash\{0\}$, in words: all scalar multiples of a vector denote the same point. We will denote the determinant

$$\begin{vmatrix} x_A & x_B & x_C \\ y_A & y_B & y_C \\ z_A & z_B & z_C \end{vmatrix}$$

corresponding to the points $A$, $B$, and $C$ by $[ABC]$. This notation is referred to as the *bracket notation*, a determinant $[ABC]$ as *a bracket*.

## 6.1 Collinearity

First, we focus on the collinearity conditions, described by $h(A, B, C)$. As stated in Section 2.2 three points $A$, $B$, and $C$ are collinear if and only if $[ABC] = 0$. This fact is made clear by the observation that

$$[ABC] = 0 \Leftrightarrow \begin{pmatrix} x_A \\ y_A \\ z_A \end{pmatrix} = \lambda \begin{pmatrix} x_B \\ y_B \\ z_B \end{pmatrix} + \mu \begin{pmatrix} x_C \\ y_C \\ z_C \end{pmatrix} \Leftrightarrow A, B, \text{ and } C \text{ are collinear.}$$

To create a proof (as shown in the next section) we need to make a connection between several conditions. This can be done using the following theorem. In order to make reading easier, we write $k$ for the point defined by coordinates $(x_k, y_k, z_k)^T$.

**Theorem 6.1.1.** *Let $1, 2, 3, 4$, and $5$ be 5 points in the plane, such that $1, 4$, and $5$ are not collinear. Then the following equivalence holds:*

$$[123] = 0 \Leftrightarrow [124][135] = [125][134]$$

**Proof** The bracket is a 3-linear alternating form: Observe

$$a = [123][145] - [124][135] + [125][134].$$

Fix $1$ in $a$. Then $a$ is a 4-linear alternating form on $\{2, 3, 4, 5\}$. Check for example what happens if we swap 3 and 5:

$$\begin{array}{rrrrrl} a' & = & [125][143] & -[124][153] & +[123][154] & \\ & = & -[125][134] & +[124][135] & -[123][145] & \\ & = & -([123][145] & -[124][135] & +[125][134] & ) \\ & = & -a. & & & \end{array} \tag{6.1}$$

However, there is no such thing as a 4-linear alternating form not equal to zero in a 3-dimensional space, so $a = 0$. Since this implies

$$[123][145] = [124][135] - [125][134]$$

the theorem is proved. □

Using this theorem, we can translate a set of conditions of the form $h(A, B, C)$ to *bi-quadratic equations*:

$$[ABD][ACE] = [ABE][ACD]$$

for any $D$ and $E$ such that $A$, $D$, and $E$ are not collinear.

This method of describing a geometry theorem implicitly introduces a number of non-degeneracy conditions. For example, the fact that 1, 4, and 5 are not collinear. On the one hand, this is an advantage, as we do not have to express this kind of non-degeneracy conditions explicitly. On the other hand, this is a disadvantage, as we might add some non-degeneracy conditions we are not aware of and which might be unnecessary. However, in this specific situation the disadvantage seems less important, since the user will construct a certain theorem in Cinderella, thus (in general) avoiding degeneration cases himself.

## 6.2 Concurrency and Conics

In this section we show how to translate the assertions $m((A, B), (C, D), (E, F))$ and $c(A, B, C, D, E, F)$ to bracket equations.

**Theorem 6.2.1.** *Observe the assertion $m((A, B), (C, D), (E, F))$. This means that the lines through $A$ and $B$, $C$ and $D$, and $E$ and $F$ go through one point. This assertion implies that all these 6 points and 3 lines are distinct, and that the point of concurrency is not one of $A, B, C, D, E, F$, thus implicitly adding non-degeneracy conditions every time we use this assertion.*

*This assertion is equivalent to,*

$$[ABC][CDE][EFA] = -[ABE][CDA][EFC],$$

*and to*

$$[ABF][CDE] = [ABE][CDF].$$

**Proof** Observe the assertion $m((A, B), (C, D), (E, F))$, i.e. the lines through $A$ and $B$, $C$ and $D$, and $E$ and $F$ go through one point, say $Z$. This is equivalent to the combination of the three assertions

$$h(A, B, Z), h(C, D, Z) \text{ and } h(E, F, Z).$$

Using Theorem 6.1.1 we find the following three equations. Notice how we have to use the fact that all 6 points are distinct and none of them is the point of concurrency.

$$\begin{aligned} [ABC][AZE] &= [ABE][AZC], \\ [CDE][CZA] &= [CDA][CZE], \\ [EFA][EZC] &= [EFC][EZA]. \end{aligned} \tag{6.2}$$

We multiply the left- and right-hand sides and cancel terms that occur on both sides, and obtain

$$[ABC][CDE][EFA] = -[ABE][CDA][EFC], \tag{6.3}$$

thus proving the first equation.

As $m((A, B), (C, D), (E, F))$ is equivalent to (for example) the assertion $m((A, B), (C, D), (F, E))$ we obtain from Equation 6.3 (the left- and right-hand side are swapped):

$$-[ABF][CDA][FEC] = [ABC][CDF][FEA]. \tag{6.4}$$

Again, we multiply the left- and right-hand sides from Equations 6.3 and 6.4 and cancel terms that occur on both sides, and we obtain

$$[ABF][CDE] = [ABE][CDF], \tag{6.5}$$

which proves the second equation of the theorem. □

We just showed two possible encodings of the $m(..)$-assertion. However, it appears that using only the second one suffices in practice. An assertion $m(..)$ gives us only three different instances of Equation 6.5, all other permutations are equivalent to one of those three.

**Remark 6.2.2.** Because of the implicit degeneration conditions introduced, we have to be careful when using $m(..)$ as a thesis assertion. Additionally, in practice it appears a proof using $m(..)$ as configuration assertions is harder to understand than a proof using $h(..)$ as configuration assertions. However, the $m(..)$-assertion still has the huge advantage that it represents three $h(..)$-assertions, thus considerably reducing the amount of configuration assertions and configuration equations.

As this shows that using the $m(..)$-assertion has both advantages we do not want to loose, and disadvantages we do not want to have, we choose to leave the choice to the user of Cinderella. When trying to obtain a proof, he can decide whether he wants to use $m(..)$-assertions in the configuration, and whether he wants to use $m(..)$-assertions in the thesis. This enables the user to find the 'golden mean' between the shortness and the clarity of the proof.

The next theorem describes how to encode six points on a conic into bracket expressions.

**Theorem 6.2.3.** *Observe the assertion $c(A, B, C, D, E, F)$. This means that the six points A,B,C,D, E, and F are on one conic. This assertion implies that all these six points are distinct and no three of the points are collinear, thus implicitly adding non-degeneracy conditions every time we use this assertion.*

*This assertion is equivalent to the following bracket equation:*

$$[ACE][BDE][ABF][CDF] = [ABE][CDE][ACF][BDF].$$

**Proof** First, observe four distinct points, $A$, $B$, $C$, and $D$, and the two degenerate conics $c_1$ and $c_2$. The conic $c_1$ is given by the line through $A$ and $B$ and the line through $C$ and $D$, the conic $c_2$ is given by the line through $A$ and $C$ and the line through $B$ and $D$. For an arbitrary point $x$ we have

$$\begin{array}{lll} x \in c_1 & \text{if and only if } x \text{ on } AB \text{ or } x \text{ on } CD\text{, so} & [ABx][CDx] = 0 \\ x \in c_2 & \text{if and only if } x \text{ on } AC \text{ or } x \text{ on } BD\text{, so} & [ACx][BDx] = 0. \end{array} \tag{6.6}$$

Now, for all $\lambda, \mu \in \mathbb{R}$, the equation

$$\lambda[ABx][CDx] + \mu[ACx][BDx]$$

describes a conic through $A$, $B$, $C$, and $D$. Now let

$$\begin{array}{rcl} \lambda & = & [ACE][BDE] \\ \mu & = & -[ABE][CDE], \end{array} \tag{6.7}$$

and observe the expression

$$[ACE][BDE][ABx][CDx] - [ABE][CDE][ACx][BDx]. \tag{6.8}$$

$$\begin{aligned}
[ABC][ADE][BDF][CEF] &= [ABD][ACE][BCF][DEF],\\
[ABE][ACD][BDF][CEF] &= [ABD][ACE][BEF][CDF],\\
[ABE][BCD][ADF][CEF] &= [ABD][BCE][AEF][CDF],\\
[ABD][AEF][BCF][CDE] &= [ABF][ADE][BCD][CEF],\\
[ABE][ACF][BDF][CDE] &= [ABF][ACE][BDE][CDF]. \qquad (6.10)
\end{aligned}$$

Table 6.1: A basis for $c(..)$-assertions in the configuration

Since this is a bi-quadratic expression of degree two, which evaluates to zero for $x \in \{A, B, C, D, E\}$, this defines a conic on the points $A$,$B$,$C$,$D$, and $E$. This means $F$ is on that conic if and only if

$$[ACE][BDE][ABF][CDF] = [ABE][CDE][ACF][BDF], \qquad (6.9)$$

which concludes the proof. □

Another proof, using Grassman-Cayley algebra, can be found in Chapter 3 of [21]. In general, a single $c(..)$-assertion gives us $6! = 720$ possible equations. However, in the configuration a basis of the subspace spanned by these 720 equations will suffice. Such a basis is a set of equations such that the other equations can be obtained by multiplying sides and removing pairs that occur on both sides. With basic linear algebra we can obtain a basis for this [22, p. 36]. The basis can be found in Table 6.1. So, every $c(..)$-assertion only adds five equations to the set of configuration equations. However, if the thesis is a $c(..)$-assertion, we have to include *all* 720 possible equations, as each of those equations is equivalent to the thesis.

## 6.3 How to Prove

Suppose we are given a certain theorem in planar geometry, containing points and some collinearity conditions. This means we know that certain brackets (i.e. expressions of the form $[ABC]$) are equal to zero. We define $B$ to be the set of all brackets, i.e. all combinations of three points from the geometry theorem, so $|B| = \binom{p}{3}$, where $p$ denotes the number of points in the configuration.

**Example** Suppose we have a configuration with the points $A$, $B$, $C$, and $D$. Then

$$B := \{[ABC], [ABD], [ACD], [BCD]\},$$

and $|B| = 4 = \binom{4}{3}$.

Suppose we have a geometry statement, and by Theorems 6.1.1, 6.2.1 and 6.2.3 we obtained a set of $n$ equations following from the configuration:

$$\begin{aligned}
c_{1l} &\equiv c_{1r},\\
c_{2l} &\equiv c_{2r},
\end{aligned}$$

$$\begin{aligned} &\vdots \\ c_{nl} \quad \equiv \quad & c_{nr}, \end{aligned} \tag{6.11}$$

where '$l$' denotes the left hand side of the equation, and '$r$' the right hand side. Each of the factors of the $c_{il}$ and $c_{ir}$ denotes a determinant of three points in the geometry statement, so each $c_{i,l/r}$ is a product of elements of $B$. Note that we use the equivalence sign '$\equiv$' rather than the normal equation sign '$=$' to make it clear that we are calculating with brackets, elements of $B$, rather than with the elements in $\mathbb{R}$ they evaluate to. From Theorem 6.1.1 it follows directly that each of the factors of the $c_{il}$ and $c_{ir}$ is not equal to zero.

Moreover, we have an (at least one) equation that implies the thesis we want to test:

$$t_l \quad \equiv \quad t_r. \tag{6.12}$$

Note that all factors in $t_{l,r}$ should be a factor of at least one $c_{i,l/r}$. This means that the brackets in the thesis equation should occur somewhere in the configuration equations.

**Remark 6.3.1.** Note that it is almost always possible to express the thesis in various different equations. For the remainder of the section we will just pick one, for ease of reading. In practice we will test all of them, checking which gives us the shortest proof, if any.

Now suppose we have a certain oracle that gives us a vector $g \in \mathbb{Q}^n$, $g \neq 0$ such that

$$\frac{1}{t_l}\prod_{i=1}^{n}(c_{il})^{g_i} \equiv \frac{1}{t_r}\prod_{i=1}^{n}(c_{ir})^{g_i}. \tag{6.13}$$

By multiplying both sides by the greatest common divisor $q$ of the denominators in $g_1, \ldots, g_n$, thus clearing the denominators, we obtain the following equation:

$$\left(\frac{1}{t_l}\right)^q \prod_{i=1}^{n}(c_{il})^{v_i} \equiv \left(\frac{1}{t_r}\right)^q \prod_{i=1}^{n}(c_{ir})^{v_i}, \text{ where } v_i = q \cdot g_i, \text{ so } v_i \in \mathbb{Z}. \tag{6.14}$$

**Remark 6.3.2.** In words: For each of the $n$ equations we multiply a certain power of the left sides with each other, and the same power of the right sides. Then all terms cancel, except for $(t_l)^q$ on the left side, and $(t_r)^q$ on the right side.

By the definition of the $c_{i,l/r}$ we know

$$(c_{il})^a \equiv (c_{ir})^a \quad \forall 1 \leq i \leq n, \forall a \in \mathbb{Z},$$

and since $q \neq 0$ we obtain from Equation 6.14:

$$\left(\frac{1}{t_l}\right)^q \equiv \left(\frac{1}{t_r}\right)^q, \quad \text{so } t_l \equiv t_r.$$

This means such a vector $g \in \mathbb{Q}^n$ gives us a *proof* that the thesis logically follows from the configuration. In the next section it will be shown how we can obtain such a $g$.

## 6.4 Obtaining the Proof

It will be shown that a vector $g$ as in 6.13 can be found by solving linear equations. We recall that $B$ is the set of all brackets, and define $b = |B|$ and $x_i$ such that $\{x_1, \dots, x_b\} = B$.

Suppose we have a configuration given by $c_{1l}, c_{1r}, \dots, c_{nl}, c_{nr}$ and a thesis given by $t_l$ and $t_r$. Recall that $c_{i,l/r}$ and $t_{l/r}$ are products of elements of $B$. Introduce the $b \times n$ matrix $X$ with coefficients in $\mathbb{Z}$, defined as follows:

$$\text{for all } 1 \le k \le b, 1 \le i \le n : X_{ki} := \begin{cases} 1 & \text{if } x_k \text{ is a factor of } c_{il}, \\ -1 & \text{if } x_k \text{ is a factor of } c_{ir}, \\ 0 & \text{otherwise.} \end{cases} \tag{6.15}$$

The vector $Y \in \mathbb{Z}^b$ is defined in the same way from our thesis.

$$\text{for all } 1 \le k \le b : Y_k := \begin{cases} 1 & \text{if } x_k \text{ is a factor of } t_l, \\ -1 & \text{if } x_k \text{ is a factor of } t_r, \\ 0 & \text{otherwise.} \end{cases} \tag{6.16}$$

Now observe the following system of linear equations

$$X \cdot g = Y, \tag{6.17}$$

with a solution vector $g$. Since $X$ and $Y$ have integer values, we know that $g \in \mathbb{Q}^n$. We will now show that $g$ satisfies Equation 6.13.

**Proof** From the fact that $g$ satisfies Equation 6.17 we find that

$$Y_k = \sum_{i=1}^{n} g_i X_{ki}, \forall 1 \le k \le b \tag{6.18}$$

so

$$\sum_{i=1}^{n} g_i X_{ki} - Y_k = 0, \forall 1 \le k \le b \tag{6.19}$$

which yields

$$\frac{1}{e^{Y_k}} \prod_{i=1}^{n} \left(e^{X_{ki}}\right)^{g_i} = 1.\forall 1 \le k \le b \tag{6.20}$$

Now

$$\text{identify } e^{Y_k} \text{ with } \begin{cases} x_k & \text{if } Y_k = 1, \\ 1/x_k & \text{if } Y_k = -1, \\ 1 & \text{if } Y_k = 0, \end{cases}$$

and

$$\text{identify } e^{X_{ki}}, 1 \le i \le n, \text{ with } \begin{cases} x_k & \text{if } X_{ki} = 1, \\ 1/x_k & \text{if } X_{ki} = -1, \\ 1 & \text{if } X_{ki} = 0, \end{cases}$$

for all $1 \le k \le b$, where $x_k \in B$. Then take the product over all $k$, and obtain

$$\frac{t_r}{t_l} \prod_{i=1}^{n} \left(\frac{c_{il}}{c_{ir}}\right)^{g_i} \equiv 1, \tag{6.21}$$

which is equal to

$$\frac{1}{t_l}\prod_{i=1}^{n} c_{il}^{g_i} \equiv \frac{1}{t_r}\prod_{i=1}^{n} c_{ir}^{g_i}, \tag{6.22}$$

which concludes the proof. □

Thus, we have a procedure that enables us to obtain a proof by solving linear equations, which is *much* faster than having to use Gröbner bases.

For the implementation there is one technicality to address.

**Remark 6.4.1.** We have to define a certain order on the brackets, to be able to conclude if two factors from the configuration equation represent the same bracket, i.e. the same element of $B$. A straightforward solution is demanding them to be ordered like the alphabet. This however introduces a small problem: When reordering the contents of certain brackets, say $[153]$ to $[135]$, the sign changes, since $[153] = -[135]$.

For each equation in the configuration we will store whether the sign is equal on both sides or not, defining

$$s_i = \begin{cases} -1 & \operatorname{sgn}(c_{il}) \neq \operatorname{sgn}(c_{ir}) \\ 1 & \operatorname{sgn}(c_{il}) = \operatorname{sgn}(c_{ir}) \end{cases} \quad (1 \le i \le n),$$

and likewise $s_0 \in \{-1, 1\}$ for the thesis.

We will then process all equations as if the sign was equal on both sides, thus making sure the equations can be handled by the procedure above. Suppose this procedure gives us a certain solution vector $g$, we simply check if the signs 'match' by testing if

$$\prod_{i=1}^{n} s_i^{v_i} = s_0^{q},$$

where $v$ and $q$ are defined as in 6.14.

If this equation holds, we have a correct proof. If it does not hold, we have a proof over the field with characteristic 2. Surprisingly, in all cases tested, the signs 'match' whenever a solution vector exists.

**Remark 6.4.2.** In general, the problem whether a geometric theorem is true or false is still equal to the decision if $t_l - t_r$ is in the ideal $I$ generated by $c_{il} - c_{ir}$ $(i = 1, \ldots, n)$. This ideal membership problem is normally decided by means of Gröbner bases, but in the previous chapter we already decided that Gröbner bases are too slow to be practical in our situation. However, a $g$ as in Equation 6.17 shows that

$$t_l - t_r \equiv \sum_{i=1}^{n} g_i (c_{il} - c_{ir}). \tag{6.23}$$

which proves the ideal membership. If such a vector $g$ does not exist however, we have no information on the ideal membership. This is why we *lose the possibility to prove a theorem to be false,* as a theorem is called false only when $t_l - t_r \notin \sqrt{I}$.

# Chapter 7

# Non-Projective Statements in Projective Geometry

In this Chapter we present the theory that enables us to represent assertions in non-projective geometry in brackets. Someone might think that calculating with expressions that are invariant with respect to linear transformations, as described in the previous chapter, automatically makes it impossible to prove any theorems containing for example circles. This is not such a strange thought, since circles might become conics (and lose their circularity) under linear transformations. However, by adding two special points to the configuration, we can prove such theorems.

Moreover, in this chapter we will make clear how to represent not just circles in bracket notation, but also perpendicularity and parallelism.

## 7.1 Complex Numbers

With the following procedure we can use complex numbers to express conditions involving distances, angles, etc. Given a point $P = (x, y) \in \mathbb{R}^2$ in the plane, we define $z_p \in \mathbb{C} := x + iy$. Moreover, $z_p = r \cdot e^{i\varphi}$ for certain $r, \varphi \in \mathbb{R}$. We know $z \in \mathbb{R} \Leftrightarrow z = \overline{z}$, and $z \in i\mathbb{R} \Leftrightarrow z = -\overline{z}$.

We switch back to homogeneous coordinates and introduce two 'points': $I = (i, -1, 0)$ and $J = (-i, -1, 0)$. Now observe the bracket $[ABI]$, where $A = (x_a, y_a, 1)$ and $B = (x_b, y_b, 1)$:

$$[ABI] = \begin{vmatrix} x_a & x_b & i \\ y_a & y_b & -1 \\ 1 & 1 & 0 \end{vmatrix} = x_a + iy_a - x_b - iy_b = z_a - z_b. \tag{7.1}$$

Likewise, the bracket $[ABJ]$ evaluates to

$$[ABJ] = \begin{vmatrix} x_a & x_b & -i \\ y_a & y_b & -1 \\ 1 & 1 & 0 \end{vmatrix} = x_a - iy_a - x_b + iy_b = \overline{z_a - z_b}. \tag{7.2}$$

**Remark 7.1.1.** Throughout this report we have been working with homogeneous coordinates, which means that if $A = (x_a, y_a, 1)$, then $A = (\lambda x_a, \lambda y_a, \lambda)$ for all $\lambda \in \mathbb{R}$, as long as $\lambda \neq 0$. Now observe what happens if we use

$$A = \begin{pmatrix} \lambda x_a \\ \lambda y_a \\ \lambda \end{pmatrix} \text{ and } B = \begin{pmatrix} \mu x_b \\ \mu y_b \\ \mu \end{pmatrix}, \text{ where } \lambda, \mu \in \mathbb{R}\backslash\{0, 1\}$$

in Equations 7.1 and 7.2. We will then find

$$[ABI] = \lambda\mu(z_a - z_b) \text{ and } [ABJ] = \lambda\mu(\overline{z_a - z_b}).$$

Thus, in order to avoid any difficulties, we will make sure whenever a bracket $[xyI]$ occurs on one side of an equation, the bracket $[xyJ]$ will occur on the other side, and vice versa.

**Example** (Collinearity) Now suppose $A$, $B$, and $C$ are collinear. Observe the complex numbers $z_1 = r_1 e^{i\varphi_1}$ of the vector $(B - A)$, and $z_2 = r_2 e^{i\varphi_2}$ of $(C - A)$. The points $A$, $B$, and $C$ are collinear if and only if the two angles $\varphi_1$ and $\varphi_2$ are either the same or opposed to each other. This means $\varphi_1 = \varphi_2$ or $\varphi_1 = \pi + \varphi_2$, which means $z_1/z_2 \in \mathbb{R}$, or equivalently

$$\frac{B - A}{C - A} = \overline{\frac{B - A}{C - A}} \tag{7.3}$$

Using Equations 7.1 and 7.2 this is equal to

$$\frac{[BAI]}{[CAI]} = \frac{[BAJ]}{[CAJ]}, \tag{7.4}$$

which evaluates to

$$[ABI][ACJ] = [ACI][ABJ], \tag{7.5}$$

which indeed fulfills the claim at Theorem 6.1.1.

## 7.2 Circles

We will now show how to encode the fact that four points are on one circle in brackets.

**Theorem 7.2.1.** *Suppose the four points $A$, $B$, $C$, and $D$ are on one circle, then the following bracket equation holds:*

$$[ACI][BDI][ADJ][BCJ] = [BCI][ADI][ACJ][BDJ].$$

*This assertion will be denoted by $ci(A, B, C, D)$.*

Note that this matches the bracket equation for a conic through the points $A$, $B$, $C$, $D$, $I$, and $J$ (See Theorem 6.2.3).

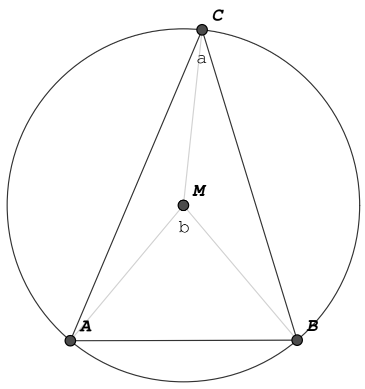

Figure 7.1:

**Lemma 7.2.2.** *Take three points in the plane, say $A$, $B$, and $C$, on a circle with midpoint $M$, angle $a = \angle BCA$ and angle $b = \angle BMA$. See Figure 7.1. First we prove that $2a = b$. Because $MA = MB = MC$ we know:*

$$\begin{array}{rcl} 2 \cdot \angle MCB + \angle CMB & = & 180 \\ 2 \cdot \angle MCA + \angle CMA & = & 180 \\ b + \angle CMB + \angle CMA & = & 360 \end{array}$$

*which yields*

$$2 \cdot \angle MCB + 2 \cdot \angle MCA - b = 0,$$

*so $2a = b$. If we move $C$ around the circle, the angle $b$ never changes, so the angle $a$ never changes. Which means, if we have a new point $D$ on the circle, we know $\angle BCA = \angle BDA$.*

*The converse is obviously true, since it is impossible to move a point $C$ away from the circle without changing $\angle BCA$.*

**Proof** We proved four points $A$, $B$, $C$, and $D$ are on one circle if and only if the angle between $AC$ and $BC$ is equal to the angle between $AD$ and $BD$. Now switch to complex numbers as described in the start of this section, and find that this is equivalent to

$$\frac{z_A - z_C}{z_B - z_C} \bigg/ \frac{z_A - z_D}{z_B - z_D} \in \mathbb{R},$$

with arguing as in the example on collinearity above. This equation can be rewritten to

$$\frac{z_A - z_C}{z_B - z_C} \bigg/ \frac{z_A - z_D}{z_B - z_D} = \overline{\frac{z_A - z_C}{z_B - z_C} \bigg/ \frac{z_A - z_D}{z_B - z_D}}.$$

Using Equations 7.1 and 7.2 this transforms to

$$[ACI][BDI][BCJ][ADJ] = [BCI][ADI][ACJ][BDJ],$$

which concludes the proof. □

A much nicer proof for Lemma 7.2.2 can be given using projective invariant theory, see for example [23] and [21].

**Proof** Observe the circle through $A$, $B$, $C$, and $D$. This is the conic through $A$, $B$, $C$, $D$, $I$, and $J$, where $I$ and $J$ are defined as above. We define a linear transformation $\pi$ that maps $(A, B, I, C)$ on $(A, B, I, D)$. Since $A$, $B$, $C$, and $D$ were on the circle, and $C$ maps to $D$, the circle remains the same.

Since the circle is left unchanged, we know that parallel lines remain parallel[1], therefore the line at infinity $l_\infty$ is mapped to $l_\infty$ under this particular transformation. Since $I$ and $J$ are (always) the incidences of the line at infinity with any circle, and $I$ is mapped to $I$, we know that $J$ has to be mapped to $J$.

Define $l$ to be the line through $A$ and $C$, and $m$ to be the line through $B$ and $C$. We know the angle between $l$ and $m$, say $\alpha$ is equal to

$$\alpha = \frac{1}{2i} \log(CR_{l_\infty}(l, m|I, J)).$$

As the cross ratio is invariant under projective transformation, we know

$$\alpha = \frac{1}{2i} \log(CR_{\pi(l_\infty)}(\pi(l), \pi(m)|\pi(I), \pi(J)).$$

Considering the fact above that $I$ is mapped to $I$, the line $l_\infty$ to $l_\infty$ and $J$ to $J$, we have

$$\alpha = \frac{1}{2i} \log(CR_{l_\infty}(\pi(l), \pi(m)|I, J),$$

which shows that the angle between $\pi(l)$ and $\pi(m)$ is equal to the angle between $l$ and $m$. Since $C$ maps to $D$ we know that $\pi(l)$ is the line through $A$ and $D$, like $\pi(m)$ is the line through $B$ and $D$. Thus, we know that $\angle ACB = \angle ADB$. $\square$

## 7.3 Parallelism and Perpendicularity

In this section we will exploit the possibilities given by the definition of $I$ and $J$, making it possible to translate assertions about, for example, parallelism to bracket equations. This might seem a strange thing to do, since parallelism is an excellent example of a non-projective assertion. However, we will show that $I$ and $J$ make it possible to represent this assertions in bracket equations, so we can use them in our prover.

**Theorem 7.3.1.** *Suppose $a$ and $b$ are two lines in the plane. Moreover, take four distinct points $A$, $B$, $C$, and $D$, such that $A$ and $B$ are incident with $a$, and $C$ and $D$ are incident with $b$. Then the assertion that $a$ and $b$ are parallel will be denoted by $par((A, B), (C, D))$ and is equivalent to*

$$m((A, B), (C, D), (I, J)).$$

[1]Take for example the line through $A$ and $B'$ and the line through $B$ and $A'$. The point $A'$ is the incidence between the circle and the line through $A$ and the midpoint $M$ of the circle, and $B'$ is the incidence between the circle and the line through $B$ and $M$.

**Proof** Since $a$ and $b$ are parallel, they meet at infinity, which is any point with homogeneous coordinates $(x, y, 0)^T$, for all $(x, y) \neq (0, 0)$. Now consider the line $l$ through $I$ and $J$. As shown in Chapter 2, this line is given by the cross product $I \times J$:

$$l = I \times J = \begin{vmatrix} e_1 & 1 & 1 \\ e_2 & i & -i \\ e_3 & 0 & 0 \end{vmatrix} = e_3 \cdot -2i = -2i(0, 0, 1)^T,$$

which means only points with a 0 as third coordinate are on this line, and all points with a 0 as third coordinate are on this line. So, the fact that $a$ and $b$ are parallel is equivalent to the fact that the lines $a$, $b$ and $l$ meet in one point. This concludes the proof. □

**Theorem 7.3.2.** *Suppose $a$ and $b$ are two lines in the plane. Take three distinct points $A$, $B$, and $C$, such that $A$ and $B$ are incident with $a$, and $A$ and $C$ are incident with $b$. Then the assertion that $a$ and $b$ are perpendicular will be denoted by $perp((A, B), (A, C))$ and is equivalent to the truth of the bracket equation*

$$[ABI][ACJ] = -[ABJ][ACI].$$

**Proof** Suppose $N = (x_N, y_N, 1)$, and write $z_N := x_N + iy_N$. Moreover, define two complex numbers,

$$z_1 := z_A - z_B = r_1 e^{\varphi_1} \text{ and } z_2 := z_A - z_C = r_2 e^{\varphi_2}.$$

The assertion that $a$ and $b$ are perpendicular through $A$ means that the lines $AB$ and $AC$ build a right angle. This is equivalent to the assertion that $\varphi_1 - \varphi_2 = \pm\frac{\pi}{2}$, so

$$\frac{z_1}{z_2} = \frac{r_1}{r_2} e^{\varphi_1 - \varphi_2} = \pm i \frac{r_1}{r_2},$$

so $\frac{z_1}{z_2} \in i\mathbb{R}$. This is equivalent to

$$\frac{z_A - z_B}{z_A - z_C} = -\overline{\frac{z_A - z_B}{z_A - z_C}},$$

which gives us the bracket expression

$$[ABI][ACJ] = -[ABJ][ACI],$$

thus concluding the proof. □

Although one might think midpoints are not so different from perpendicularity and parallelism, we found out it is much harder to translate into bracket equations. In fact, up to now we haven't found an encoding for this assertion.

Chapter 8

# On the Implementation

In this chapter some notes are given on the implementation of the prover described in Chapters 6 and 7. This chapter does not include any source code, it is purely meant to give an *overview* of how the transition from a geometric theorem in Cinderella to a proof of that theorem can be realized.

## 8.1 Translating the Assertion

The translation from a geometric theorem in Cinderella into a set of assertions of the forms described in Chapter 6 and 7, takes place in two steps. These two steps correspond to the translations from Item 1 to 2, and from Item 2 to 3 in Section 8.2.

Firstly, an algorithm in Cinderella (see for example Section A) is translated into an OpenMath `plangeo.assertion`-object, as described in Section 4.2. This can be done rather straightforward, as every step of the algorithm corresponds to a single OpenMath Application. For example,

```
a := Join(A,B)
```

will be translated into

```
<OMA>
  <OMS name="line" cd="plangeo1"/>
  <OMV name="a"/>
  <OMA>
    <OMS name="incident" cd="plangeo1"/>
    <OMV name="a"/>
    <OMV name="A"/>
  </OMA>
  <OMA>
    <OMS name="incident" cd="plangeo1"/>
    <OMV name="a"/>
    <OMV name="B"/>
  </OMA>
</OMA>
```

directly. Thus, walking through Cinderella's algorithm step by step, we obtain an OpenMath `plangeo1.assertion` representing the configuration. The thesis added to this object is the last non-trivial incidence the randomized prover concluded.

Cinderella is able to convert the following Cinderella algorithms into OpenMath elements: `Join`, `Meet`, `Mid`, `PointOnLine`[1], `Through`[2], `Orthogonal`, `Parallel`, `CircleMP`[3], `ConicBy5`, `IntersectionConicLine`, `IntersectionConicConic`, `CircleBy3`, `PointOnCircle`, `OtherIntersectionCC`[4], and `OtherIntersectionCL`[5]. Note that not all of these algorithms can be encoded to bracket equations. However, it is useful to translate as much objects as possible to OpenMath, as this OpenMath object can be used by other applications.

In this step we will restrict ourselves to elements we can translate to the asertions given in Chapters 6 and 7. This means that if the OpenMath object from the previous step has elements such as `Mid`, an error will be raised and the translation will be broken off at this point. However, the OpenMath object is still valid, and might be used by an application that can handle more statements. Moreover, this two-phased design makes it possible for the prover to handle any theorem in projective geometry that can be expressed in OpenMath, not just the ones that can be constructed in Cinderella!

The `plangeo1.assertion` from the previous step is processed in the following way:

1. Find all elements (point, lines, conics) in the configuration, say $\{E_1, \ldots, E_p\}$,

2. Find all incidences, and link them to the elements, thus finding a set of incidences $F_i \subset \{E_1, \ldots, E_p\}$, where $1 \leq i \leq p$. This means that element $E_i$ is incident to all elements in the set $F_i$,

3. We set $G := \{1, \ldots, p\}$,

4. While $G \neq \emptyset$:

    (a) Get an index $k \in G$, where first indices corresponding to conics are processed, then circles, then points, and finally lines,

    (b) Encode the element identified by $k$. For example: If $E_k$ is a conic, and $F_k$ contains 6 points, an element of the form $c(..)$ is added to the configuration,

    (c) Set $G := G \backslash \{k\}$

    (d) Set $F_i := F_i \backslash \{E_k\}$, for all $i \in G$,

    (e) Set $G := G \backslash \{i\}$ for all $i \in G$ for which $F_i = \emptyset$.

[1]A new point on an existing line
[2]A new line through an existing point
[3]MP stands for MidPoint
[4]The two intersections between two conics
[5]The two intersections between a conic and a line

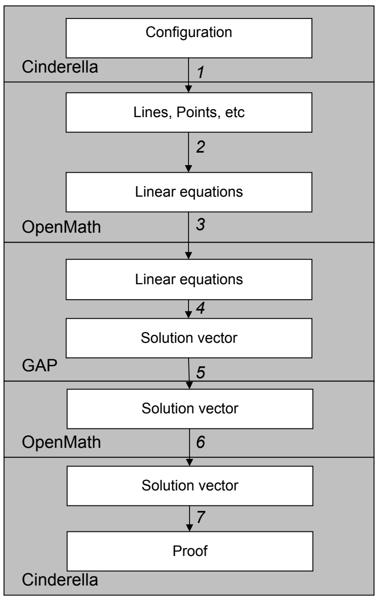

Figure 8.1: The chain of translations

5. Find the incidence the thesis describes. Depending on if this is an incidence between a point and a line, a point and a conic or a point and a circle, the type of the thesis will differ.

Notice that the element 'points' in Item 4a may be ignored if the user stated that he does not want $m(..)$ assertions in the configuration (See Remark 6.2.2). Using the above procedure, a configuration from Cinderella is translated automatically to a configuration consisting of assertions, ready to be converted into brackets.

## 8.2 The Prover

The procedure explained in the previous sections was implemented in Cinderella [4], with the aid of OpenMath [12, 13, 15] and GAP[20].

We cover the following chain of translations, the first 2 items of which are identical to the translations described in Section 4.2. See Figure 8.1 for a schematic overview.

1. **Cinderella**: A configuration described by points and lines, some objects may have coordinates,
2. **OpenMath**: A configuration described by points and lines, as in `plangeo`, some objects may have coordinates,
3. **OpenMath**: A matrix $X$ and a vector $Y$ as in Equations 6.15 and 6.16, respectively,

4. **GAP**: A matrix $X$ and a vector $Y$ as in Equations 6.15 and 6.16, respectively,
5. **GAP**: A vector $v$ as in Equation 6.13,
6. **OpenMath**: A vector $v$ as in Equation 6.13,
7. **Cinderella**: A vector $v$ as in Equation 6.13,
8. **Cinderella**: A string representing the proof, where the coordinates of the vector $v$ have been translated back into their equivalents in bracket expressions.

These translations were implemented using Java, the Riaca OpenMath Library[15] and GAP[20]. The result was integrated within Cinderella and will be a part of Cinderella 2, which will be ready someday in the future with more exciting new options!

# Chapter 9

# Examples

In this chapter we give some examples of geometric theorems proved using Cinderella and GAP. These theorems were created in Cinderella, 'discovered' by the internal Randomized Prover, and then, via OpenMath, given to the prover. Thus, there has been no optimization whatsoever by the user.

## 9.1 Pappos revisited

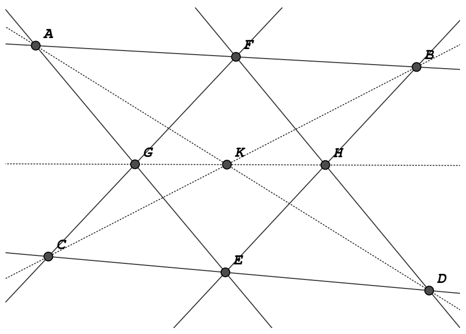

Figure 9.1: Pappos' Theorem

We again consider a version Pappos' Theorem, see Figure 9.1. The thesis is that the lines through $B$ and $C$, through $A$ and $D$, and through $G$ and $H$ go through one point ($K$). The full output of the prover is as follows:

```
Conditions:
{h(C, F, G)} On line h
{h(D, F, H)} On line g
{h(B, E, H)} On line f
{h(A, E, G)} On line e
{h(C, D, E)} On line c
{h(A, B, F)} On line a

Assertion:
{m((B, C), (A, D), (G, H))} Through point K

Number of configuration equations: 200
Number of possible theses: 3
```

```
Found a proof for thesis 1. Length: 11.
Found a proof for thesis 2. Length: 11.
Found a proof for thesis 3. Length: 10.

(1) [A.C.F][B.C.G] ==  [B.C.F][A.C.G] <== {h(C, F, G)}
(1) [B.D.F][A.D.H] ==  [A.D.F][B.D.H] <== {h(D, F, H)}
(1) [B.C.E][A.B.H] ==  [A.B.E][B.C.H] <== {h(B, E, H)}
(1) [A.B.E][B.D.H] ==  [B.D.E][A.B.H] <== {h(B, E, H)}
(1) [A.B.E][A.C.G] ==  [A.C.E][A.B.G] <== {h(A, E, G)}
(1) [A.D.E][A.B.G] ==  [A.B.E][A.D.G] <== {h(A, E, G)}
(1) [B.C.D][A.C.E] ==  [A.C.D][B.C.E] <== {h(C, D, E)}
(1) [A.C.D][B.D.E] ==  [B.C.D][A.D.E] <== {h(C, D, E)}
(1) [A.B.C][A.D.F] ==  [A.B.D][A.C.F] <== {h(A, B, F)}
(1) [A.B.D][B.C.F] ==  [A.B.C][B.D.F] <== {h(A, B, F)}
---------------------------------------------------------------------
(1) [B.C.G][A.D.H] ==  [B.C.H][A.D.G] <== {m((B, C), (A, D), (G, H))}

Checking proof... done.
Result: [B.C.G][A.D.H] == [A.D.G][B.C.H]

Found first proof in 2.08 seconds, final proof in 3.27 seconds.
```

In the remainder of the section only the proofs are given, the rest of the prover output is omitted. A shorter proof can be obtained when using $m(..)$-assertions in the configuration:

```
(1) [A.B.D][C.G.H] == -[A.B.H][C.D.G] <== {m((A, B), (D, H), (C, G))}
(1) [C.D.G][A.B.H] == -[A.C.D][B.G.H] <== {m((C, D), (A, G), (B, H))}
---------------------------------------------------------------------
(1) [A.B.D][C.G.H] ==  [A.C.D][B.G.H] ==> {m((B, C), (A, D), (G, H))}
```

## 9.2 Pascal

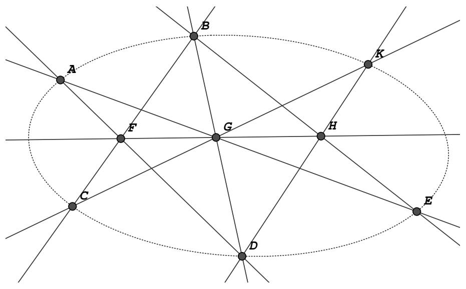

Figure 9.2: Pascal's Theorem

We consider Pascal's Theorem, as shown in the picture above. The thesis is that the six points $A$, $B$, $C$, $D$, $E$ and $K$ are on one conic. The proof is as follows:

```
(1)                [A.C.G][B.C.K] ==  [B.C.G][A.C.K]              <== {h(C, G, K)}
(1)                [B.D.H][A.D.K] ==  [A.D.H][B.D.K]              <== {h(D, H, K)}
(1)                [B.F.G][A.F.H] ==  [A.F.G][B.F.H]              <== {h(F, G, H)}
(1)                [B.C.G][A.B.F] ==  [A.B.C][B.F.G]              <== {h(B, C, F)}
(1)                [A.B.C][B.F.H] ==  [B.C.H][A.B.F]              <== {h(B, C, F)}
(1)                [A.B.D][A.F.G] ==  [A.D.G][A.B.F]              <== {h(A, D, F)}
(1)                [A.D.H][A.B.F] ==  [A.B.D][A.F.H]              <== {h(A, D, F)}
(1)                [A.B.E][B.C.H] ==  [B.C.E][A.B.H]              <== {h(B, E, H)}
(1)                [B.D.E][A.B.H] ==  [A.B.E][B.D.H]              <== {h(B, E, H)}
(1)                [A.C.E][A.B.G] ==  [A.B.E][A.C.G]              <== {h(A, E, G)}
(1)                [A.B.E][A.D.G] ==  [A.D.E][A.B.G]              <== {h(A, E, G)}
-------------------------------------------------------------------------------------------
(1) [A.C.E][A.D.K][B.C.K][B.D.E] ==  [B.D.K][B.C.E][A.D.E][A.C.K] <== {co(A, B, C, D, E, K)}
```

## 9.3 Desargues

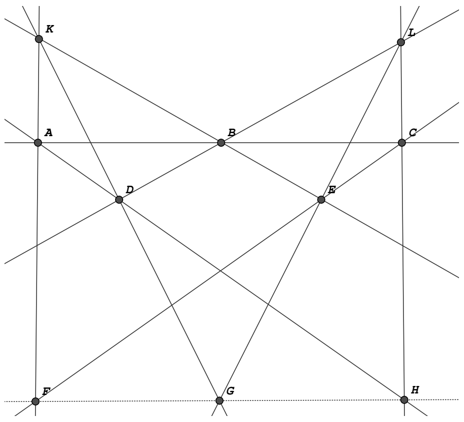

Figure 9.3: Desargues' Theorem

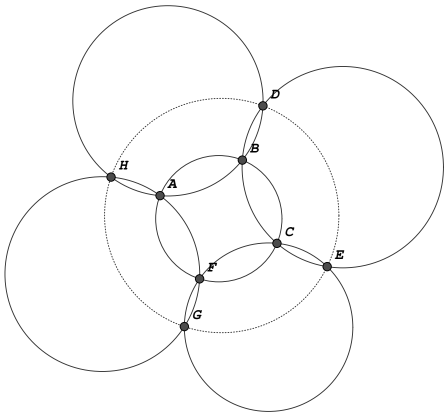

Figure 9.4: Miguel's six circle theorem

Desargues states that $F$, $G$ and $H$ are on one line. This can be shown by the prover as follows:

```
(1) [B.C.D][E.G.H] == -[B.D.H][C.E.G] <== {m((C, H), (B, D), (E, G))}
(1) [A.B.E][D.F.G] == -[B.E.F][A.D.G] <== {m((A, F), (B, E), (D, G))}
(1) [A.C.D][B.D.H] ==  [A.B.D][C.D.H] <== {h(A, D, H)}
(1) [A.D.G][C.D.H] ==  [A.C.D][D.G.H] <== {h(A, D, H)}
(1) [C.E.G][A.E.F] ==  [A.C.E][E.F.G] <== {h(C, E, F)}
(1) [A.C.F][B.E.F] ==  [B.C.F][A.E.F] <== {h(C, E, F)}
(1) [A.B.F][A.C.H] ==  [A.B.H][A.C.F] <== {h(A, B, C)}
(1) [A.B.H][A.C.K] ==  [A.B.K][A.C.H] <== {h(A, B, C)}
(1) [A.B.K][A.C.E] ==  [A.B.E][A.C.K] <== {h(A, B, C)}
(1) [A.B.D][B.C.F] ==  [A.B.F][B.C.D] <== {h(A, B, C)}
------------------------------------------------------------------
(1) [D.F.G][E.G.H] ==  [E.F.G][D.G.H] <== {h(F, G, H)}
```

## 9.4 Miguel

Miguel states that if $ABCF$, $BCDE$, $CEFG$, $AFGH$ and $ABDH$ form five circles, then the four points $D$, $E$, $G$ and $H$ are on one circle, see Figure 9.4. This is proved as follows:

```
(1) [A.F.I][F.G.H][A.H.J][G.I.J] ==  [A.F.H][F.G.I][A.I.J][G.H.J] <== {ci(A, F, G, H)}
(1) [A.F.H][A.I.J][F.G.J][G.H.I] ==  [A.F.J][A.H.I][F.G.H][G.I.J] <== {ci(A, F, G, H)}
(1) [C.E.I][C.F.J][E.G.J][F.G.I] ==  [C.E.J][C.F.I][E.G.I][F.G.J] <== {ci(C, E, F, G)}
(1) [B.C.I][B.D.J][C.E.J][D.E.I] ==  [B.C.J][B.D.I][C.E.I][D.E.J] <== {ci(B, C, D, E)}
(1) [A.B.H][B.D.I][A.I.J][D.H.J] ==  [A.B.I][B.D.H][A.H.J][D.I.J] <== {ci(A, B, D, H)}
(1) [A.B.J][A.H.I][B.D.H][D.I.J] ==  [A.B.H][A.I.J][B.D.J][D.H.I] <== {ci(A, B, D, H)}
(1) [A.B.I][B.C.F][A.F.J][C.I.J] ==  [A.B.F][B.C.I][A.I.J][C.F.J] <== {ci(A, B, C, F)}
(1) [A.B.F][A.I.J][B.C.J][C.F.I] ==  [A.B.J][A.F.I][B.C.F][C.I.J] <== {ci(A, B, C, F)}
------------------------------------------------------------------------------------------
(1) [D.E.I][D.H.J][E.G.J][G.H.I] ==  [G.H.J][E.G.I][D.H.I][D.E.J] <== {ci(D, E, G, H)}
```

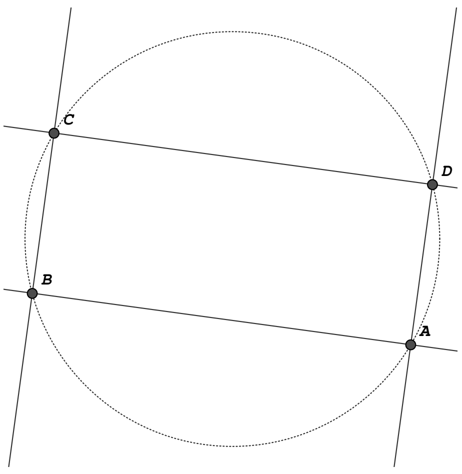

Figure 9.5: A rectangle and a circle

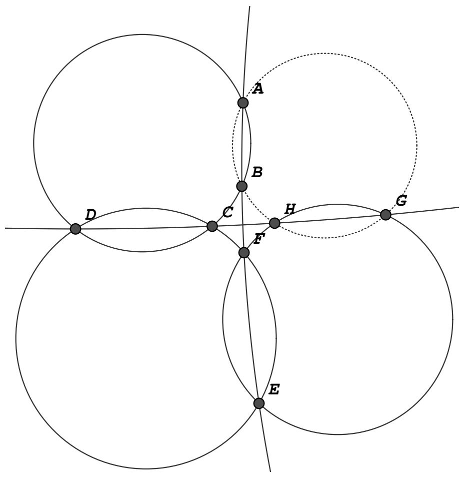

Figure 9.6: Another six circle theorem

## 9.5 A rectangle

In Figure 9.5 an example can be found of a theorem containing parallel and perpendicular lines. Our prover gives us the following proof for this theorem:

```
(2)            [B.C.J][A.B.I] == -[B.C.I][A.B.J]           <== {perp((B, C), (B, A))}
(1)            [C.D.I][A.B.J] ==  [C.D.J][A.B.I]           <== {par((C, D), (A, B))}
(1)            [A.D.J][B.C.I] ==  [A.D.I][B.C.J]           <== {par((A, D), (B, C))}
----------------------------------------------------------------------------------------
(1) [A.B.I][A.D.J][B.C.J][C.D.I] ==  [C.D.J][B.C.I][A.D.I][A.B.J] ==> {ci(A, B, C, D)}
```

## 9.6 Six circles

Observe the geometric configuration in Figure 9.6. The thesis is that the points $A$, $B$, $G$ and $H$ are on one circle. Although this theorem is a lot like Miguel's theorem about six circles, thisone can not be proved to be true by our prover. In [22] Umbach gives a proof in 4 steps, using human reasoning about this configuration. He too, however, is unable to prove this theorem automatically.

# Chapter 10

# Conclusion

The goal of this project was to make it possible to proof geometric theorems put together in a geometry program. The most obvious way to automatically prove geometric theorems is using Gröbner bases. A few weeks into this project, it became clear that the Gröbner bases algorithm is too slow to prove geometric theorems whose polynomial representation is not tweaked. That is when we decided to use the method described in Chapter 6.

Considering Chapter 7, one might say that projective geometry with $I$ and $J$ actually *is* Euclidian geometry. It looks like it could be possible to translate all possible configurations that one might build in Cinderella to bracket algebra. However, we must be careful not to forget that we do not have the possibility to prove theorems false, as we did when using Gröbner bases. Secondly, the prover can proof *some* theorems in non-projective geometry, but often fails, as for example in Section 9.6. Moreover, we traded the common polynomial representation for a bracket notation most people are unfamiliar with, though it is not hard to learn. But, in exchange for these disadvantages, we gained the possibility to proof geometric theorems considerably faster than before. Moreover, these proofs are short and, unlike proofs made by Gröbner bases, easy to check by hand.

In the course of the project the power of OpenMath became clear. Because of the existing link between OpenMath and GAP [15] it was extremely easy to use GAP for solving linear equations without any additional programming. Although that is not such a difficult algorithm, there is no need to implement it yourself. The added advantage is that GAP will perform a lot better than our own home-made algorithm. Other advantages of the extensive use of OpenMath include the possibility to reuse intermediate results, import geometric theorems made by hand, and in the future, use other computer algebra packages than GAP, or import geometric theorems made by other geometry programs. All this can be done rather easily, because of the clarity of the OpenMath standard.

A few questions remain open on the use of the bracket notation and proving geometry using statements that are invariant under projective transformations. Personally, I regret that I did not have more time to look into these problems. Firstly, we again consider the fact that, when proving a geometric theorem, we are actually testing ideal membership (see Remark 6.4.2). One would expect the restriction to linear combinations to be such a huge limitation that the ideal membership can hardly ever be proved. In practice however, we are able to find a proof for almost every geometric theorem considered. It would be very interesting to find out why we 'get lucky' so often. This might give us the key as to why we can prove Miguel's theorem (Section 9.4) and fail to prove the six circle theorem in Section 9.6. Moreover, it would be interesting to investigate why theorems containing non-projective geometry fail to be proved more often than theorems that contain only projective geometry statements. Other things that might be interesting to consider are how to obtain a proof of minimal length, how to minimize the amount of configuration equations passed to GAP, etc. In short: there are many interesting things to be done in this field.

Cinderella and its randomized prover have been out there for a few years already, the invariant theory used exists since the twenties, and solving linear equations is not the most recent discovery either. However, the combination of these three items into a single program gives us a rather interesting result. OpenMath made it possible to do this in a structured and extendable manner, helping to achieve the goal of this project. It is now finally possible to create geometric theorems by pointing and clicking, and then automatically obtain a proof for that theorem. Not only can this proof be obtained very quickly, it is short and easy to check!

# Appendix A

# Algorithms

## A.1 Pappos' Theorem

The capital characters denote the points, the small characters denote the lines.

```
A := FreePoint;
B := FreePoint;
a := Join(A,B);
C := FreePoint;
b := Join(B,C);
D := FreePoint;
c := Join(C,D);
E := FreePoint;
d := Join(D,E);
F := Meet(a,c);
e := Join(E,F);
G := Meet(b,d);
f := Join(A,G);
g := Join(A,D);
h := Join(B,E);
H := Meet(e,f);
k := Join(H,C);
```

# Appendix B

# Systems of Polynomials

## B.1 Pappos' Theorem - 1

$$
\begin{aligned}
c_1 &= la_x \cdot pA_x + la_y \cdot pA_y + la_z \cdot pA_z, \\
c_2 &= la_x \cdot pB_x + la_y \cdot pB_y + la_z \cdot pB_z, \\
c_3 &= lb_x \cdot pB_x + lb_y \cdot pB_y + lb_z \cdot pB_z, \\
c_4 &= lb_x \cdot pC_x + lb_y \cdot pC_y + lb_z \cdot pC_z, \\
c_5 &= lc_x \cdot pC_x + lc_y \cdot pC_y + lc_z \cdot pC_z, \\
c_6 &= lc_x \cdot pD_x + lc_y \cdot pD_y + lc_z \cdot pD_z, \\
c_7 &= ld_x \cdot pD_x + ld_y \cdot pD_y + ld_z \cdot pD_z, \\
c_8 &= ld_x \cdot pE_x + ld_y \cdot pE_y + ld_z \cdot pE_z, \\
c_9 &= pF_x \cdot la_x + pF_y \cdot la_y + pF_z \cdot la_z, \\
c_{10} &= pF_x \cdot lc_x + pF_y \cdot lc_y + pF_z \cdot lc_z, \\
c_{11} &= le_x \cdot pE_x + le_y \cdot pE_y + le_z \cdot pE_z, \\
c_{12} &= le_x \cdot pF_x + le_y \cdot pF_y + le_z \cdot pF_z, \\
c_{13} &= pG_x \cdot lb_x + pG_y \cdot lb_y + pG_z \cdot lb_z, \\
c_{14} &= pG_x \cdot ld_x + pG_y \cdot ld_y + pG_z \cdot ld_z, \\
c_{15} &= lf_x \cdot pA_x + lf_y \cdot pA_y + lf_z \cdot pA_z, \\
c_{16} &= lf_x \cdot pG_x + lf_y \cdot pG_y + lf_z \cdot pG_z, \\
c_{17} &= lg_x \cdot pA_x + lg_y \cdot pA_y + lg_z \cdot pA_z, \\
c_{18} &= lg_x \cdot pD_x + lg_y \cdot pD_y + lg_z \cdot pD_z, \\
c_{19} &= lh_x \cdot pB_x + lh_y \cdot pB_y + lh_z \cdot pB_z, \\
c_{20} &= lh_x \cdot pE_x + lh_y \cdot pE_y + lh_z \cdot pE_z, \\
c_{21} &= pH_x \cdot le_x + pH_y \cdot le_y + pH_z \cdot le_z, \\
c_{22} &= pH_x \cdot lf_x + pH_y \cdot lf_y + pH_z \cdot lf_z, \\
c_{23} &= lk_x \cdot pH_x + lk_y \cdot pH_y + lk_z \cdot pH_z, \\
c_{24} &= lk_x \cdot pC_x + lk_y \cdot pC_y + lk_z \cdot pC_z, \\
c_{25} &= pK_x \cdot lh_x + pK_y \cdot lh_y + pK_z \cdot lh_z, \\
c_{26} &= pK_x \cdot lk_x + pK_y \cdot lk_y + pK_z \cdot lk_z.
\end{aligned}
\tag{B.1}
$$

## B.2 Pappos' Theorem - 2

$$\begin{aligned}
c_1 &= pA_x \cdot (pB_y \cdot pF_z - pF_y \cdot pB_z) - pB_x \cdot (pA_y \cdot pF_z - pF_y \cdot pA_z) \\
&\quad + pF_x \cdot (pA_y \cdot pB_z - pB_y \cdot pA_z), \\
c_2 &= pE_x \cdot (pG_y \cdot pD_z - pD_y \cdot pG_z) - pG_x \cdot (pE_y \cdot pD_z - pD_y \cdot pE_z) \\
&\quad + pD_x \cdot (pE_y \cdot pG_z - pG_y \cdot pE_z), \\
c_3 &= pE_x \cdot (pF_y \cdot pH_z - pH_y \cdot pF_z) - pF_x \cdot (pE_y \cdot pH_z - pH_y \cdot pE_z) \\
&\quad + pH_x \cdot (pE_y \cdot pF_z - pF_y \cdot pE_z), \\
c_4 &= pC_x \cdot (pD_y \cdot pF_z - pF_y \cdot pD_z) - pD_x \cdot (pC_y \cdot pF_z - pF_y \cdot pC_z) \\
&\quad + pF_x \cdot (pC_y \cdot pD_z - pD_y \cdot pC_z), \\
c_5 &= pA_x \cdot (pH_y \cdot pG_z - pG_y \cdot pH_z) - pH_x \cdot (pA_y \cdot pG_z - pG_y \cdot pA_z) \\
&\quad + pG_x \cdot (pA_y \cdot pH_z - pH_y \cdot pA_z), \\
c_6 &= pG_x \cdot (pC_y \cdot pB_z - pB_y \cdot pC_z) - pC_x \cdot (pG_y \cdot pB_z - pB_y \cdot pG_z) \\
&\quad + pB_x \cdot (pG_y \cdot pC_z - pC_y \cdot pG_z), \\
c_7 &= pA_x \cdot (pK_y \cdot pD_z - pD_y \cdot pK_z) - pK_x \cdot (pA_y \cdot pD_z - pD_y \cdot pA_z) \\
&\quad + pD_x \cdot (pA_y \cdot pK_z - pK_y \cdot pA_z), \\
c_8 &= pE_x \cdot (pK_y \cdot pB_z - pB_y \cdot pK_z) - pK_x \cdot (pE_y \cdot pB_z - pB_y \cdot pE_z) \\
&\quad + pB_x \cdot (pE_y \cdot pK_z - pK_y \cdot pE_z). \qquad \text{(B.2)}
\end{aligned}$$

# Bibliography

[1] Thomas Becker and Volker Weispfennig. *Gröbner Bases, A Computational Approach to Commutative Algebra*, volume 141 of *Graduate Texts in Mathematics*. Springer-Verlag, New York, 1993.

[2] S-C. Chou. Automated reasoning in geometries using the characteristic set method and gröbner basis method. In *Proceedings of the international symposium on Symbolic and algebraic computation*, pages 255–260. ACM Press, 1990.

[3] S-C. Chou and X-S. Gao. Methods for mechanical geometry formula deriving. In *Proceedings of the international symposium on Symbolic and algebraic computation*, pages 265–270. ACM Press, 1990.

[4] The Interactive Geometry Software Cinderella. `http://www.cinderella.de`.

[5] David Cox, John Little, and Donal O'Shea. *Using algebraic geometry*, volume 185 of *Graduate Texts in Mathematics*. Springer-Verlag, New York, 1998.

[6] G.-M. Greuel, G. Pfister, and H. Schönemann. SINGULAR 2.0. A Computer Algebra System for Polynomial Computations, Centre for Computer Algebra, University of Kaiserslautern, 2001. `http://www.singular.uni-kl.de`.

[7] Deepak Kapur. Geometry theorem proving using hilbert's nullstellensatz. In *Proceedings of the fifth ACM symposium on Symbolic and algebraic computation*, pages 202–208. ACM Press, 1986.

[8] Deepak Kapur. Using grob̈ner bases to reason about geometry problems. *Journal of Symbolic Computation*, 2(4):399–408, 1986.

[9] Ulrich Kortenkamp. *Foundations of Dynamic Geometry*. PhD thesis, Swiss Federal Institute of Technology, Zurich, 1999. `http://www.cinderella.de/papers/diss.pdf`.

[10] Martin Kreuzer and Lorenzo Robbiano. *Computational commutative algebra. 1*. Springer-Verlag, Berlin, 2000.

[11] B. Kutzler and S. Stifter. On the application of buchberger's algorithm to automated geometry theorem proving. *Journal of Symbolic Computation*, 2(4):389–397, 1986.

[12] OpenMath. `http://www.openmath.org`.

[13] OpenMath Content Dictionary: plangeo1..5. This CD defines symbols for planar Euclidean geometry. `http://www.win.tue.nl/~amc/oz/om/cds/geometry.html`.

[14] OpenMath Content Dictionary: polygb2. This CD contains operators for Groebner basis computations with polynomial expressions. `http://www.win.tue.nl/~amc/oz/om/cds/polygb2.html`.

[15] The Riaca OpenMath Library. `http://www.riaca.win.tue.nl`.

[16] Jürgen Richter-Gebert. Mechanical theorem proving in projective geometry. *Annals of Mathematics and Artificial Intelligence*, 13:139–172, 1995.

[17] Jürgen Richter-Gebert and Ulrich H. Kortenkamp. *The Interactive Geometry Software Cinderella, Version 1.2, Manual*. Berlin, 2000.

[18] Dan Roozemond. Bachelorproject - Automatic Geometric Theorem Proving. (A copy of this report can be obtained by sending an e-mail to d.a.roozemond@student.tue.nl), 2003.

[19] Dan Roozemond and Stefan van Zwam. Algebra 3 eindopdracht, 'algebraic numbers and the three-colorability of graphs'. A copy of this report can be obtained by sending an e-mail to either d.a.roozemond@student.tue.nl or s.h.m.v.zwam@student.tue.nl, 2002.

[20] Martin Schönert et al. *GAP – Groups, Algorithms, and Programming*. Lehrstuhl D für Mathematik, Rheinisch Westfälische Technische Hochschule, Aachen, Germany, fifth edition, 1995.

[21] Bernd Sturmfels. *Algorithms in Invariant Theory*. Springer-Verlag, Wien, New York, 1993.

[22] Andreas Umbach. Automatisches erzeugen geometrischer beweise. Master's thesis, Institut für Theoretische Informatik ETH Zürich, 2000.

[23] Ulrich H. Kortenkamp und Jürgen Richter-Gebert. Euklidische und Nicht-Euklidische Geometrie in Cinderella. `http://www.cinderella.de/papers/nichtEuklidisch.pdf`.

# Index