Formalization of Wu's simple method in Coq

Jean-David Génevaux, Julien Narboux, Pascal Schreck

Formalization of Wu's simple method in Coq

Jean-David Génevaux, Julien Narboux, and Pascal Schreck

LSIIT UMR 7005 CNRS - Université de Strasbourg*

Abstract. We present in this paper the integration within the Coq proof assistant, of a method for automatic theorem proving in geometry. We use an approach based on the validation of a certificate. The certificate is generated by an implementation in Ocaml of a simple version of Wu's method.

Keywords: formalization, automation, geometry, Wu's method, proof assistant, Coq

1 Introduction

Bringing new automation techniques to interactive theorem provers while preserving safety is an important goal in order to spread the use of proof assistants.

In this paper, we focus on one of the most successful methods for automated theorem proving in geometry, namely Wu's method. We integrate this method into the Coq proof assistant [Coq10]. Indeed, Wu's method can prove hundreds of geometry theorems [Wu78] and some conjectures were first proved using this method [Cho88,Wan89].

One could formally prove correct the implementation of this decision procedure within a proof assistant such as Coq or Isabelle, but this would require a tedious work. Another approach is to modify the decision procedure to make it produce a witness of its correctness (the certificate) which can be checked by an external tool called a validator. The certificate is such that the validator is simpler to prove than the original decision procedure. Using such an approach, the completeness of the method is not guaranteed as the certifying algorithm may generate an erroneous certificate or even fail to generate one. But if we prove formally the correctness of the validator, when the validator confirms a result we get the same level of confidence as if we had proven the correctness of the decision procedure. This approach has several advantages: first the algorithm which generates the certificate is independent of the validator and hence can be written by different people using different languages, second the decision procedure does not need to be proved formally, hence the implementation can be optimized more easily. The difficulty is to generate a certificate which can be checked in a reasonable amount of time.

Gröbner basis is another well known tool for automated deduction in geometry [Kap86]. Gröbner basis have already been integrated into Coq by Benjamin

* This work is partially supported by the ANR project Galapagos.

Grégoire, Loïc Pottier and Laurent Théry [Pot08,GPT11]. Formalizing Wu's method brings us a tool to decide ideal membership as the Gröbner basis method does. But, in the context of automatic deduction in geometry, Wu's method and its variants developed by Shang-Ching Chou, Xiao-Shan Gao and Dongming Wang have been shown to be generally more efficient [CG90,Wan01,Wan04]. Moreover, in geometry, the statements as given by a user are usually false in some special cases, such as when three points are collinear for example. These conditions under which a statement is true are called non degeneracy conditions (ndgs). An advantage of Wu's method is that it can *generate* the ndgs. This is crucial for the applications we aim at in the context of education. In the future, we would like to integrate this method in a tool to prove geometry theorems within a dynamic geometry software following the work presented in [Pha10,PBN11].

1.1 Related Work

Wu's method has been implemented by Chou, he used his implementation to prove hundreds of geometric problems [Cho88]. Xiao-Shan Gao created Geometry Expert (GEX) [GL02,Gao00] and Zheng Ye produced Java Geometry Expert (Java GEX) [YCG11]. Dongming Wang developed further elimination algorithms based on pseudo-division and implemented them as a Maple library called Geother [Wan02]. Judit Robu implemented Wu's method in Theorema [Rob02]. Predrag Janičić implemented the method within GCLC [JQ06].

We are not aware of any formalization of Wu's method inside a proof assistant. But other methods for automatic deduction in geometry have been integrated into proof assistants: Gröbner basis can be used in Coq, HOL-Light and Isabelle [GPT11,Har07,CW07], the area method for non oriented euclidean geometry has been formalized in Coq as a tactic [CGZ94,Nar04,JNQ10] and geometric algebras are also available in Coq and can be used to prove automatically theorems in projective geometry [FT11].

The closest work to ours is the implementation of a Gröbner basis method within Coq by Benjamin Grégoire, Loïc Pottier and Laurent Théry [GPT11]. In this paper, we present an extension of their work to deal with Wu's method.

This paper is organized as follows. We first give a quick overview of Wu's method and we highlight the facts we need to prove to show the correctness of the method. Then we describe our formalization within the Coq proof assistant.

2 Overview of Wu's Method

2.1 Cartesian geometry

We are interested in formally proving theorems in geometry, but there are many geometries! Wu's method focuses on geometry with coordinates in a field. To be more precise, we consider here the plane $\mathbb{F}^2$ where $\mathbb{F}$ is a field where all predicates on points, lines, circles, etc. can be expressed under the form $P(X) = 0$, X being a vector of variables corresponding to the coordinates of the involved objects.

Let us give a simple (and very classical) example:

Example 1.

In a parallelogram the diagonals intersect in their midpoints:

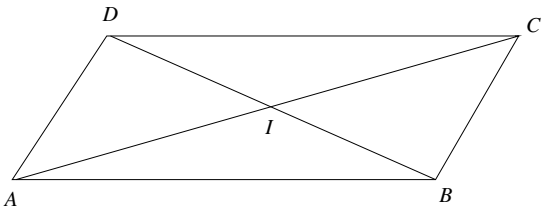

Fig. 1. Classical parallelogram example

In the usual Euclidean geometry, the fact that lines (AB) and (CD) are parallel can be expressed by the polynomial equation[1]

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

The fact that point I is the midpoint of $[AC]$ and the midpoint of $[BD]$ can be expressed by the polynomial equations $(x_B + x_D) - (x_A + x_C) = 0$ and $(y_B + y_D) - (y_A + y_C) = 0$.

The theorem stating that in a parallelogram the diagonals intersect in their midpoints is then directly stated by

$$\begin{cases} (x_B - x_A)(y_C - y_D) - (y_B - y_A)(x_C - x_D) = 0 \\ (x_C - x_B)(y_D - y_A) - (y_C - y_B)(x_D - x_A) = 0 \end{cases}$$

$$\Rightarrow$$

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

where all the variables are implicitly universally quantified. □

The goal of Wu's method is precisely to prove geometric theorems which can be put under this algebraic form.

2.2 Rings and ideals

The general question is the following:

[1] Note that this equation is actually equivalent to $A = B$ or $C = D$ or (AB) and (CD) are parallel. See the discussion on ndgs (Sec.3).

(1) Given k polynomials $H_1, \ldots, H_k$ and G in $\mathbb{F}(x_1, \ldots x_m)$, is it true that

$$\forall x_1, \ldots x_m \in \mathbb{F} : \bigwedge_{i=1}^{k} H_i(x_1, \ldots x_m) = 0 \;\Rightarrow\; G(x_1, \ldots, x_m) = 0 \; ?$$

This question gets obviously a positive answer when G belongs to the radical of the ideal $I =< H_1, \ldots, H_k >$ generated by polynomials $H_1, \ldots, H_k$, that is, when there exists an integer r and k polynomials $Q_1, \ldots, Q_k$ such that $G^r = \Sigma_i Q_i H_i$. The famous Hilbert's *Nullstellensatz* theorem states that if $\mathbb{F}$ is algebraically closed, then the converse is also true. That is, we can always find such polynomials.

Then, in this framework, proving a geometric theorem consists in proving that the polynomial belongs to an ideal for which a set of generator is known. Note that the *Nullstellensatz* has to be used only to prove that a polynomial does not belong to an ideal: if we want to prove a known theorem, producing a linear combination of polynomials H_i equal to G is enough.

It would be easy to test ideal membership by using Euclidean division. But, unfortunately, the ring of multivariate polynomials over a field is not Euclidean when more than one variable is involved. Nonetheless, the two most famous methods use a kind of Euclidean division to perform such a test. Buchberger's method consists in transforming the generating set into a so-called Gröbner basis in which a division algorithm can be used to give the good result while in the Wu's method a *pseudo-division* is used which closely mimics the Euclidean division.

2.3 Pseudo-division and pseudo-remainder

The idea of pseudo-division consists in multiplying the polynomial to be divided, say P, by the leading coefficient of the dividing polynomial, say Q, to a certain power in order that all the coefficients of this product are divisible by the leading coefficient of Q. More formally, let P and Q be polynomials of $\mathbb{F}[x_1, \ldots, x_m]$, and $I \in \mathbb{F}[x_1, \ldots, x_{m-1}]$ be the leading coefficient of Q in the variable x_m, the pseudo-division of P by Q in the variable x_m yields two polynomials T and R such that

$$I^r P = TQ + R$$

where r is the number of non zero coefficients of P and $R \in \mathbb{F}[x_1, \ldots, x_m]$ with $\deg(x_k, R) < \deg(x_k, Q)$. Establishing the existence of the polynomials T and R and the correction of the pseudo-division algorithm is very similar to the proof of the correction of the Euclidean division algorithm. R is called the pseudo-remainder of P by Q into the variable x_k and it obviously belongs to the ideal $< P, Q >$.

Pseudo-division and pseudo-remainder can be used to perform a triangulation of the system $\bigwedge_i H_i(x_1, \ldots, x_m) = 0$. The triangulation produces an algebraic system $\bigwedge_i T_i(x_1, \ldots, x_m) = 0$ where each T_i belongs to $\mathbb{F}[x_1, \ldots x_i] \cap <$

$H_1, \ldots, H_k >$. This way, the implication

$$\bigwedge_i H_i(x_1, \ldots, x_m) = 0 \Rightarrow \bigwedge_i T_i(x_1, \ldots, x_m) = 0$$

holds, but the converse is true if and only if the leading coefficient considered in the used pseudo-divisions do not vanish. But for the formalization, we do not the need the converse.

Note that, if the statement is given as a ruler-and-compass construction then the triangulation is tractable. Moreover, after choosing a reference, it may even become trivial. For instance, by fixing point A at $(0, 0)$ and point B on Ox axis and observing that $y_C = y_D$, the statement of example 1 becomes :

$$(x_C - x_B - x_D).y_C = 0 \Rightarrow x_B + x_D - x_C = 0$$

where x_B, x_C and y_C are *parameters* of the figure and x_D is a dependent variable. Note that, by making this choice, we implicitly assume that $A \neq B$. The hypotheses are trivially under a triangular form. Next Section explains in details how to fix a *reference*, why this is correct, and how to construct the figure corresponding to the hypotheses.

Eventually, using successive pseudo-divisions of G by $T_k, T_{k-1}, \ldots, T_1$, the method tests if IG belongs to the ideal generated by polynomials H_i where I is a product of the leading coefficients of the triangulated polynomials. Summarizing the calculi, Wu's method, in his simplest form, allows to compute some polynomials I, S_i and R such that:

$$IG = \sum_i (H_i S_i) + R$$

It is clear that if the polynomial R is null, then the theorem is proved under the assumption that $I(x_1, \ldots, x_m) \neq 0$. When the theorem is stated as a construction this assumption corresponds to the non degeneracy conditions [CG92].

In our example, the degenerated case occurs when $y_C = 0$. This matches the case where the four points A, B, C and D are collinear : in this case, the theorem does not hold. The converse is not true: $R \neq 0$ does not mean that the theorem is false. The simple method of Wu is not complete but according to Chou [Cho88] it is powerful enough to prove hundred of classical theorems. A complete method would need to use ascending chains which are considered in the Ritt-Wu principle [Cho88].

The main steps of the Wu's method for geometry theorem proving can be summarized as follows:

1. Transforming the statements into an algebraic form.
2. Choosing the origin and a direction for the system of coordinates.
3. Showing that to prove the statement in general it is sufficient to show that it holds in the given system of coordinates.
4. Simplifying the polynomials thanks to the choice of the reference.
5. Triangulating the list of hypotheses using pseudo-division.

6. Pseudo-dividing successively the goal by the triangulated hypotheses. If the final remainder is the null polynomial, then the statement holds under the condition that some polynomials do not vanish (we call those polynomials the non degeneracy conditions)
7. Re-interpreting non degeneracy conditions as geometric predicates.

3 Formalization in Coq

In this section we describe our formalization of Wu's method in Coq. To formalize a decision procedure within Coq, there are several solutions:

1. We could write a tactic in the implementation language of Coq (Ocaml) which generates a proof term.
2. We could write a tactic in Ltac, the tactic language of Coq: a domain specific language which allows pattern matching on the context of the proof and backtracking.
3. We could implement the decision procedure within Coq itself, prove its correctness and use it within Coq using reflection.
4. As explained in the introduction we can implement the decision procedure as an external tool which generates a certificate and then check the certificate by reflection using a procedure written in Coq itself.

The first and third solutions can be seen as special (extreme) cases of the fourth one. For the first solution, the certificate can be considered as the proof itself and the validator is trivial, and for the third solution the certificate contains no information and the validator is the whole decision procedure.

As explained in the introduction for the core of the method we choose an approach based on the validation of a certificate, and we have been able to reuse the machinery developed by Benjamin Grégoire, Loïc Pottier and Laurent Théry for the Gröbner basis. But for some steps of the method we use the tactic language of Coq (Ltac). We use Ltac for the first steps, to put the problem in algebraic form and to choose a system of coordinates. Figure 2 gives an overview of the different parts of the development and their implementation language. The simplification phase consists only perfoming some small simplifications using substitution in Ltac before starting the actual procedure when equations are trivial. The geometrization phase is not yet implemented. The simple procedure of Wu that we use may produce ndgs which are not necessary. We plan to implement geometrization only for the predicates corresponding to ndgs conditions of ruler and compass constructions (collinear, parallel, point equality).

In the next sections, we describe first the algebraization process and then the generation of the certificate. We do not describe fully the validator here because we could reuse the validator of Benjamin Grégoire, Loïc Pottier and Laurent Théry.

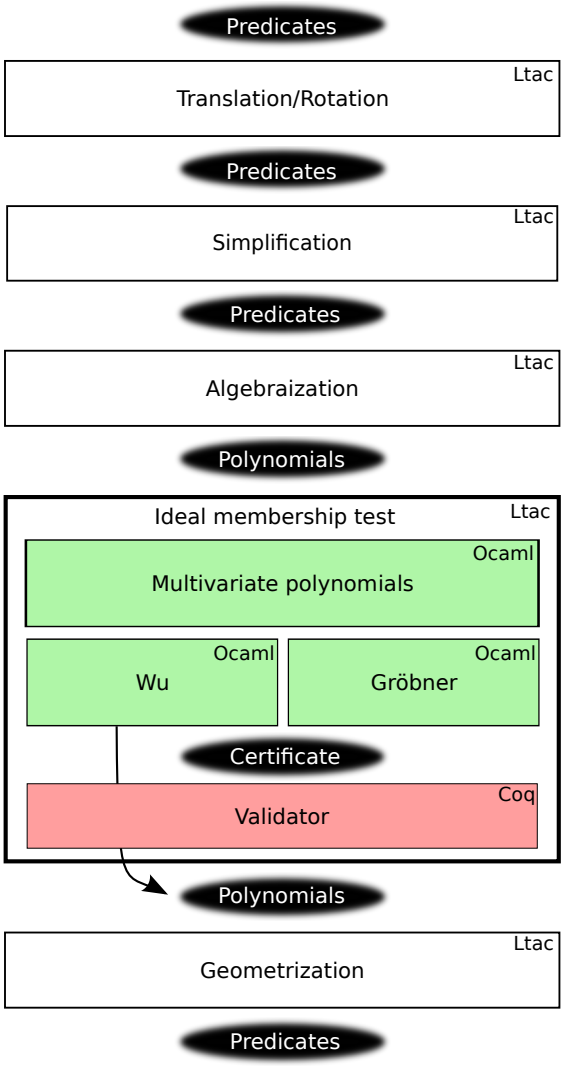

Fig. 2. Overview of the development

3.1 Algebraization

Putting the statement in algebraic form, choosing the right system of coordinates, and simplifying the generated coordinates, is the first step of the method, and it is crucial. In this section we describe the formalization of this step using the tactic language of Coq.

Stating the conjecture As most automatic theorem provers in geometry, to state a theorem, we will assume that the user provides the assumptions as a list of geometric predicates taking only points as parameters. This has the advantage to simplify the formalization and as shown in [Nar07], it is possible to transform a statement containing points and lines into a statement containing only points. Figure 3 gives the Coq definitions of some common geometric predicates. `X A` and `Y A` denotes respectively the x and y coordinates of point A. Figure 3 lists some of the Coq definitions for the common geometric predicates. These definitions have the advantage that degenerated cases are not excluded. For instance, `parallel A B C D` holds when $A = B$. This leads to statements which are as general as possible.

```
Definition collinear  A B C :=
  (X A - X B) * (Y B - Y C) = (Y A - Y B) * (X B - X C).
Definition parallel A B C D :=
  (X A - X B) * (Y C - Y D) = (Y A - Y B) * (X C - X D).
Definition orthogonal A B C D :=
  (X A - X B) * (X C - X D) + (Y A - Y B) * (Y C - Y D) = 0.
Definition is_midpoint I A B :=
  2 * X I = X A + X B /\ 2 * Y I = Y A + Y B.
Definition length_eq A B C D :=
  (X A - X B) * (X A - X B) + (Y A - Y B) * (Y A - Y B) =
  (X C - X D) * (X C - X D) + (Y C - Y D) * (Y C - Y D).
Definition is_in_intersection A B C D E :=
  collinear A B C /\ collinear A D E.
```

Fig. 3. Definition of some geometric predicates

Then the statement corresponding to our example (without the ndg) is the following:

```
Lemma parallelogram : forall A B C D E F:Point,
  B <> A ->
  parallel A B C D ->
  parallel A D B C ->
  is_midpoint E A C ->
  is_midpoint F B D ->
  equal E F.
```

To put the statement in algebraic form we just need to unfold the definitions of the geometric predicates. But in practice, even for simple examples, if we do not fix a specific coordinate system the computations will take much more time or may even fail due to lack of memory. The classical solution consists in adding additional assumptions to fix the coordinate system, assuming that one point is the origin and another belongs to the x axis:

```
Lemma parallelogram : forall A B C D E F:Point,
  X A = 0 -> Y A = 0 -> Y B = 0 ->
  B <> A ->
  parallel A B C D ->
  parallel A D B C ->
  is_midpoint E A C ->
  is_midpoint F B D ->
  equal E F.
```

This solution is sufficient to perform benchmarks in a context where the user is an expert. As we aim to apply this method in an educational context, we want to provide a procedure which works on the original statement.

We need to show that, *without loss of generality*, we can assume that A is of coordinates $(0, 0)$ and B of coordinates $(x_b, 0)$. Following the idea of John Harrison [Har09], this requires to show that the predicates we use are invariant under translation and rotation. For that purpose, we define a translation function taking as arguments a point and a vector (`trans`) and a rotation function (`rot`) taking as arguments a point and the sine and cosine of the angle of rotation. Then for each predicate (collinear in this example), we prove lemmas of the following form:

```
Lemma collinear_inv_translation: forall A B C V,
 collinear A B C <->
 collinear (trans A V) (trans B V) (trans C V).
Lemma collinear_inv_rotation: forall A B C cos sin,
 cos*cos + sin*sin = 1 ->
 (collinear A B C <->
  collinear (rot A cos sin) (rot B cos sin) (rot C cos sin)).
```

Finally, we have a tactic `Algebraization O I H` which takes as input a point O to be put at the origin, a point I to be put on the x-axis and a proof that $O \neq I$, and which makes use of the above lemmas to perform the required simplifications. The algebraization tactic only work for goals which are stated using function and predicates which take as arguments only points. The following predicates/functions are available: collinear, parallel, orthogonal, midpoint, intersection of lines, square of length, equality of points, angles or lengths. The tactic can not deal with user defined predicates automatically. Adding a new predicate requires to add the lemmas for invariance under translation and rotation and to update the tactic.

We leave for future work, the automatic choice of a reference. Our experiments shows that this choice is crucial : choosing different references can lead to

very different computation times. The heuristic proposed by Chou is to choose as an axis a line which contains many points and also a perpendicular line. We noticed that choosing as the origin one of the points of the goal is also often a good choice, because it simplifies the starting polynomial of the successive pseudo-division.

After application of the tactic `Algebraization A B H`, we get the following goal:

```
H : 0 = - X P2 * P3 + X P0 * P3
H0 : - P3 * X P0 + Y P2 * X P0 = X P2 * Y P0
H4 : 2 * X P = X P0
H5 : 2 * Y P = P3 + Y P0
H2 : 2 * X P1 = X P2
H6 : 2 * Y P1 = Y P2
______________________________________(1/1)
X P = X P0
```

3.2 Certified implementation

In the section we describe our implementation of a simple version of Wu's method and how it generates certificates.

Implementation of the method To integrate Wu's method in Coq, we need to modify the implementation to generate a certificate. Unfortunately, the implementations of Wu's method which are already available are either not open source or rely on proprietary Computer Algebra Systems. Moreover, we hope that in the future our tactic could be distributed with Coq, hence we decided to write our own version of the method in Ocaml, the implementation language of Coq. We base our implementation of Wu's method on the Ocaml libraries for dealing with multivariate polynomials developed by Loïc Pottier for the integration of Gröbner basis in Coq. He used two different data structures for polynomials : first a list of monomials, each monomial is represented by a coefficient and an array containing the degree of each variable and the total degree of the monomial, second a recursive data structure considering polynomials with several variables as a polynomial with one variable but whose coefficients are polynomials in the other variables. We reused his data-structures, and slightly optimized the one based on list of monomials by changing the representation of monomials: in the arrays of degrees we store only the degrees of the variables until the highest variable which appears in the monomial. This variant reduces both the memory footprint and the number of needed comparisons of degree. This is crucial, because 30% of the computing time is spent in the monomial comparison function which is the core of the multiplication function. We implement the simple algorithm presented by Chou in [Cho88], hence it is incomplete because we do not check polynomials for irreducibility. But in practice many theorems can be proved using this simple method. Our certifying implementation consists in 3000 lines of Ocaml half of which are tests and examples.

Certificate generation Certificates are pieces of information, which allow to check if the result of a computation is correct. In our case, the certificate we generate is based on the *Nullstellensatz*, hence we need to produce a proof that the polynomial which represents the goal (multiplied by a coefficient representing the ndgs) is in the ideal generated by the hypotheses. We need to show that there are some integer r, polynomials I and S_i such that:

$$I^r G = \sum_i (S_i H_i)$$

As explained in Sec. 2, the core of Wu's method can be decomposed into two main steps: the triangulation and the successive pseudo-divisions.

To generate the certificate we generate three kinds of intermediary certificates: one for the pseudo-divisions, one for the triangulation, and another one for the successive pseudo-division procedure.

Certificates for pseudo-division The certificates for the pseudo-division will be used both by the triangulation and successive pseudo-division procedure.

Recall that the pseudo-division of A and B return a remainder R[2] such that there exists an integer d and a polynomial Q such that:

$$I^d A = QB + R$$

where I is the leading coefficient of B.

So to certify that R is in the ideal generated by A and B, instead of just returning the remainder R we also return I, d and Q which verify that $R = I^d \times A + (-Q) \times B$.

To ease implementation we number our polynomials. Then the implementation of the pseudo-division is encapsulated into a function which stores in the list of certificates a certificate for each polynomial identified by its number. The certificate for each polynomial is in the form of a list of pairs of a polynomial and the number of a polynomial which we already know to be in the ideal :

```
let pseudo_div_num a b x certif =
    let (r,c,d,s) = pseudo_div (a.p) (b.p) x in
    let new_n = new_num () in
      certif := (new_n, r, [(c^^d, a.n);(p_zero -- s, b.n)])
                ::(!certif);
      {p=r ; n= new_n}
```

Certificates for the triangulation phase For the triangulation, we do not need to show that the result is triangulated, we just need to show that the polynomials in the triangulation are in the ideal generated by the hypotheses. We notice that the triangulation phase of Wu's method and its variants based on pseudo-division

[2] Note that there is a condition on the degree of R but this is not important in this discussion.

rely on an invariant: the method maintains a list of polynomials which belong to the ideal generated by the hypotheses. At the beginning of the triangulation procedure, the list consists of the original hypotheses themselves, hence they are in the ideal. At each step of the triangulation, the list of polynomials l is replaced by a new list l' such that l and l' differ only in one polynomial $p \in l$ (let's call it $p' \in l'$). This polynomial p' is in the ideal generated by l.

The fact that p' is in the ideal generated by l follows from the fact that p' is computed using a pseudo-division $p' = prem(p, h)$ for some h in l.

Hence, to generate the certificate of the triangulation, we keep track of every intermediate polynomial thanks to its number, and we combine the certificates of the pseudo-divisions.

Certificates for the successive pseudo-division For the last phase (the successive pseudo-division of the goal by the triangulated polynomials), we just save the S_i, and I as:

$$IG = \sum_i (S_i T_i)$$

where $S_i = q_i \prod_{j=1}^{i-1} c_j^{d_j}$ and $I = \prod_{j=1}^{n} c_j^{d_j}$ and the c_j are the leading coefficients of the polynomials in the triangulation.

But, if during the combination of the different auxiliary certificates we evaluate the S_i, then we get polynomials with thousands of terms. Polynomials of this size cannot be managed by Coq. Grégoire, Pottier and Théry proposed to use certificates involving not simply the list of the S_i but a *straight line program* to evaluate the S_i. This provides the possibility to have some `let ... in` instructions inside the certificates to allow to decompose and to share polynomial expressions. The validator is a function which computes IG and $\sum_i (S_i T_i)$ and which test the equality of these two polynomials.

Using their certificate structure we can define auxiliary polynomials which can then be used to define several other polynomials, *etc.*

To reduce the time required to check certificates, we make use of those `let ... in` instructions as most as possible. But as shown in the Sec. 4 the verification time of the certificates is still significant.

Toward more optimized certificates for ideal membership tests In this section we present some ideas for improvement (which are not yet implemented) of the structure for certificates.

One limitation of the certificates we use relies on the fact that every polynomial which is defined by a `let ... in` needs to be in the ideal generated by the previous polynomials. This bring no restriction for the polynomials generated by the Gröbner basis method, but in the context of Wu's method, it would be useful to be able to define polynomials using `let ... in` which are not in the ideal generated by the hypotheses. This would allow some sharing between the definition of I and the definition of the S_i. Another idea to reduce drastically the size of the certificates would be to allow to have unevaluated pseudo-divisions in the certificates. This would require to prove formally the pseudo-division. But

the pseudo-division is a time consuming task of Wu's method, hence this would imply to do a lot of computations during the validation of the certificate.

4 Benchmark

Table 1 presents our results compared to those of L. Pottier *et al.* based on Gröbner basis. The results show no clear winner, Wu's method is faster in some cases and Gröbner basis is faster in some other cases. We were surprised by the time needed to check the certificate in Coq. To study this, we reimplemented the validator in Ocaml and used it with our implementation of the method. The first column provides the computation time to produce the certificate *and* to check it when using an Ocaml validator : those timing are encouraging. The percentage of time required to check the certificates using the Ocaml validator varies from 1% to 80% but it is about 50% on average.

We did not expect such a difference of computation time between Ocaml and Coq. The data structure used by the Coq validator is a a modified Hörner form provided by the ring tactic[GM05]. We tried another implementation based on lists of monomials, but the computation time was similar. The results could be improved in the future by the use of Native-Coq a Coq version using the native Ocaml compiler to perform strong reduction [BDG11].

Table 1. Benchmark : Computations are done using an Intel(R) Core(TM) i5 CPU 750 @ 2.67GHz with 4Gb RAM.

Theorem	Certificate generated using **Wu** and checking using **Ocaml**	Certificate generated using **Wu** and checking using **Coq**	Certificate generated using **GB** and checking using **Coq**	Relative speed of Wu's method over GB
Pascal_2	0.013	21	-	-
Pascal_1	0.024	22	1652	×75
Ptolemy95	0.010	10	30	×3
Pappus	0.043	3	8	×2.6
Altitudes	0.002	3	7	×2.3
Simson	0.002	5	8	×1.6
Perp-bisect	0.001	2	3	×1.5
Pythagore	0.001	1	1	×1
Feuerbach	0.038	15	15	×1
Isoceles	0.001	1	1	×1
Euler Line	0.063	9	6	×0.6
Medians	0.001	3	2	×0.6
Chords	0.015	4	2	×0.5
Thales	0.003	6	3	×0.5
Bissectors	0.001	6	3	×0.5
Desargues	0.027	99	10	×0.1
Ceva	0.025	98	6	×0.06

5 Conclusion

In this paper, we described the first formalization of Wu's simple method in an interactive proof assistant. We implemented a version of Wu's method which generates certificates to ensure the correctness of the results generated by the method. We reused and extended the framework for certificate validation introduced by Benjamin Grégoire, Loïc Pottier and Laurent Théry, and we present some ideas for improvement. Thanks to the power of Wu's method, the new tactic we obtain does not only prove geometry theorems but also generates non degeneracy conditions necessary to prove a conjecture. Compared to other implementations of the method, as we formalize the invariance by translation and rotation of the statements, we prove the real geometry theorems and not a special case of the algebraic version of the statement given in a specific coordinate system.

In the future, we plan to extend our OCaml implementation with a more optimized version of the pseudo-division (using GCD computations) along with the complete method of Wu-Ritt [Wu78,Cho88,CG90] or its variants developed by Dongming Wang [Wan01,Wan04]. For educational applications our implementation still needs to be polished for example by extending the input language and by choosing automatically an origin and a direction for the system of coordinates.

The formalization of Wu's method in Coq enriches our framework for interactive and automatic theorem proving in geometry. We have now four methods for automated deduction in geometry available in Coq. It opens the door to several directions of research. Among them, it would be interesting to study the combination of Wu's method with other methods inside the same framework, for instance with the cylindrical algebraic decomposition method (CAD) whose formalization in Coq is under way by Assia Mahboubi [Mah06]. We could also study the different applications of Wu's method, such as solving geometric constraint systems [GC98a,GC98b] or finding loop invariants in the context of program verification.

Availability. The current version of the plug-in is available here:
`http://dpt-info.u-strasbg.fr/~narboux/nsatzwu.tar.gz`

Acknowledgments. We wish to thank Loïc Pottier and Laurent Théry for having made their work publicly available and for the discussions we had.

References

[BDG11] Mathieu Boespflug, Maxime Dénès, and Benjamin Grégoire. Full reduction at full throttle. In *First International Conference on Certified Programs and Proofs CPP 2011, Taiwan, December 7-9*, Lecture Notes in Computer Science. Springer, 2011. To Appear.

[CG90] Shang-Ching Chou and Xiao-Shan Gao. Ritt-Wu's Decomposition Algorithm and Geometry Theorem Proving. In Mark E. Stickel, editor, *CADE*, volume 449 of *Lecture Notes in Computer Science*, pages 207–220. Springer, 1990.

[CG92] Shang-Ching Chou and Xiao-Shan Gao. A class of geometry statements of constructive type and geometry theorem proving. In *Proceeding of CADE 92*, 1992.

[CGZ94] Shang-Ching Chou, Xiao-Shan Gao, and Jing-Zhong Zhang. *Machine Proofs in Geometry*. World Scientific, Singapore, 1994.

[Cho88] Shang-Ching Chou. *Mechanical Geometry Theorem Proving*. D. Reidel Publishing Company, 1988.

[Coq10] Coq development team, The. *The Coq proof assistant reference manual, Version 8.3*. LogiCal Project, 2010.

[CW07] Amine Chaieb and Makarius Wenzel. Context aware calculation and deduction. In Manuel Kauers, Manfred Kerber, Robert Miner, and Wolfgang Windsteiger, editors, *Calculemus/MKM*, volume 4573 of *Lecture Notes in Computer Science*, pages 27–39. Springer, 2007.

[FT11] Laurent Fuchs and Laurent Théry. A Formalisation of Grassmann-Cayley Algebra in Coq. In *Post-proceedings of Automated Deduction in Geometry (ADG 2010)*, 2011.

[Gao00] Xiao-Shan Gao. Geometry expert, software package, 2000.

[GC98a] Xiao-Shan Gao and Shang-Ching Chou. Solving geometric constraint systems. I. A global propagation approach. *Computer Aided Design*, 30(1):47–54, 1998.

[GC98b] Xiao-Shan Gao and Shang-Ching Chou. Solving geometric constraint systems. II. A symbolic approach and decision of Rc-constructibility. *Computer Aided Design*, 30(2):115–122, 1998.

[GL02] Xian-Shan Gao and Qiang Lin. MMP/Geometer - a software package for automated geometry reasoning. In F. Winkler, editor, *Proceedings of Automated Deduction in Geometry (ADG 2002)*, Lecture Notes in Computer Science, pages 44–46. Springer-Verlag, 2002.

[GM05] Benjamin Grégoire and Assia Mahboubi. Proving Equalities in a Commutative Ring Done Right in Coq. In *Theorem Proving in Higher Order Logics (TPHOLs 2005)*, volume 3603 of *Lecture Notes in Computer Science*, pages 98–113. Springer, 2005.

[GPT11] Benjamin Grégoire, Loïc Pottier, and Laurent Théry. Proof Certificates for Algebra and their Application to Automatic Geometry Theorem Proving. In *Post-proceedings of Automated Deduction in Geometry (ADG 2008)*, number 6701 in Lecture Notes in Artificial Intelligence, 2011.

[Har07] John Harrison. Automating elementary number-theoretic proofs using gröbner bases. In Frank Pfenning, editor, *Proceedings of the 21st International Conference on Automated Deduction, CADE 21*, volume 4603 of *Lecture Notes in Computer Science*, pages 51–66, Bremen, Germany, 2007. Springer-Verlag.

[Har09] John Harrison. Without loss of generality. In Stefan Berghofer, Tobias Nipkow, Christian Urban, and Makarius Wenzel, editors, *TPHOLs*, volume 5674 of *Lecture Notes in Computer Science*, pages 43–59. Springer, 2009.

[JNQ10] Predrag Janičić, Julien Narboux, and Pedro Quaresma. The Area Method: a Recapitulation. *Journal of Automated Reasoning*, 2010. online first.

[JQ06] Predrag Janičić and Pedro Quaresma. System Description: GCLCprover + GeoThms. In *Automated Reasoning*, volume 4130 of *Lecture Notes in Artificial Intelligence*, pages 145–150. Springer-Verlag, 2006.

[Kap86] Deepak Kapur. Geometry Theorem Proving using Hilbert's Nullstellensatz. In *SYMSAC '86: Proceedings of the fifth ACM symposium on Symbolic and algebraic computation*, pages 202–208, New York, NY, USA, 1986. ACM Press.

[Mah06] Assia Mahboubi. *Contributions à la certification des calculs dans R : théorie, preuves, programmation*. PhD thesis, Université de Nice Sophia-Antipolis, Nov 2006.

[Nar04] Julien Narboux. A Decision Procedure for Geometry in Coq. In Slind Konrad, Bunker Annett, and Gopalakrishnan Ganesh, editors, *Proceedings of TPHOLs'2004*, volume 3223 of *Lecture Notes in Computer Science*. Springer-Verlag, 2004.

[Nar07] Julien Narboux. A graphical user interface for formal proofs in geometry. *J. Autom. Reasoning*, 39(2):161–180, 2007.

[PBN11] Tuan Minh Pham, Yves Bertot, and Julien Narboux. A Coq-based Library for Interactive and Automated Theorem Proving in Plane Geometry. In *Proceedings of the 11th International Conference on Computational Science and Its Applications (ICCSA 2011)*, Lecture Notes in Computer Science. Springer-Verlag, 2011.

[Pha10] Tuan Minh Pham. An Additional Tool About the Orientation for Theorem Proving in the Coq Proof Assitant. In *Proceedings of Automated Deduction in Geometry (ADG 2010)*, 2010.

[Pot08] Loïc Pottier. Connecting Gröbner Bases Programs with Coq to do Proofs in Algebra, Geometry and Arithmetics. In G. Sutcliffe, P. Rudnicki, R. Schmidt, B. Konev, and S. Schulz, editors, *Knowledge Exchange: Automated Provers and Proof Assistants*, CEUR Workshop Proceedings, page 418, Doha, Qatar, 2008.

[Rob02] Judit Robu. *Geometry Theorem Proving in the Frame of the Theorema Project*. PhD thesis, Johannes Kepler Universität, Linz, September 2002.

[Wan89] Dongming Wang. A new theorem discovered by computer prover. *Journal of Geometry*, 36:173–182, 1989. 10.1007/BF01231031.

[Wan01] Dongming Wang. *Elimination Method*. Springer-Verlag, 2001.

[Wan02] Dongming Wang. GEOTHER 1.1: Handling and Proving Geometric Theorems Automatically. In *Proceedings of Automated Deduction in Geometry (ADG 2002)*, volume 2930 of *Lecture Notes in Computer Science*, pages 194–215, 2002.

[Wan04] Dongming Wang. *Elimination Practice*. Springer-Verlag, 2004.

[Wu78] Wen-Tsün Wu. On the Decision Problem and the Mechanization of Theorem Proving in Elementary Geometry. In *Scientia Sinica*, volume 21, pages 157–179. 1978.

[YCG11] Zheng Ye, Shang-Ching Chou, and Xiao-Shan Gao. An Introduction to Java Geometry Expert. In *Post-proceedings of Automated Deduction in Geometry (ADG 2008)*, volume 6301, pages 189–195. Springer-Verlag, 2011.