Available online at www.sciencedirect.com

ScienceDirect

Electronic Notes in Theoretical Computer Science 338 (2018) 61–77

Electronic Notes in Theoretical Computer Science

www.elsevier.com/locate/entcs

Formalizing Constructive Projective Geometry in Agda

Guillermo Calderón[1]

University of the Republic
Montevideo, Uruguay

Abstract

We present a formalization of Projective Geometry in the proof assistant and programming language Agda. We formalize a recent development in constructive Projective Geometry which has been shown appropriate to cover most traditional topics in the area applying only constructively valid methods. The equivalence with other well-known constructive axiomatic systems for projective geometry is proved and formalized.
The implementation covers a basic fragment of intuitionistic synthetic Projective Plane Geometry including the axioms, principle of duality, Fano and Desargues properties and harmonic conjugates.
We focus in an illustrative example of implementation of a complex and large proof which appears partially and vaguely described in the literature; namely the *uniqueness of the harmonic conjugate*.
The most important details of our implementation are described and we show how to take advantage of several interesting properties of Agda such as *modules*, *dependent record types*, *implicit arguments and instances* which result very helpful to reduce the typical verbosity of formal proofs.

Keywords: proof assistants, formalizations of mathematics, projective geometry, type theory, Agda

1 Introduction

Projective Geometry is a well-established branch of mathematics which studies the incidence properties of points, lines and planes. Typical textbooks relative to this area (e.g. [4,23]) cover topics such as: axiomatisation of points and lines and of the incidence relation, the principle of duality, the theoremes of Desargues and Fano, harmonic conjugates, projectivities, polarities, conics, etc..

Projective geometry constitutes a very elegant, self-contained mathematical system. It is constructed from two primitive concepts: points and lines together with a relation of incidence among them. In addition, a few axioms determine the behaviour of the entire system. For this reason, projective geometry becomes a very attractive discipline to be formalized in a computer system. and an interesting case study in order to investigate problems involved with computer formalization of mathematics.

[1] Email: calderon@fing.edu.uy

https://doi.org/10.1016/j.entcs.2018.10.005

Despite its simplicity, projective geometry is considered as a unifying framework for all other geometries. Every result in projective geometry can be applied to affine geometry which in turn reduces to Euclidean geometry. This feature is implicit in the so called Erlangen program of F. Klein [9] where projective geometry is defined as the study of the properties invariant under projectivities.

On the other hand, projective geometry has a number of important applications in different areas of computer science, such as: computer vision, cryptography, computer graphics [3,6,2]. A computer formalization of projective geometry will contribute to solve known problems in these areas.

This paper describes a computer formalization of *constructive projective plane geometry*. In a first instance, we implement the system of axioms presented in Mandekern's paper [15] which is based on previous constructive developments of projective geometry such as [7] and [22].

Our formalization [2] is written in *Agda* [19], a proof assistant and functional programming language based on *Intuitionistic Type Theory* [17]. In particular, we are concerned with the construction of a programming tool which helps us to write formal proofs in projective geometry in a comfortable way but without using automated tactics.

The constructive method implies among other things that the principle of excluded middle is not used at all. In particular, the decidability of the basic relations (notably the equality) is not assumed.

Although the use of a constructive approach requires much more additional effort than the necessary in non constructive approaches, we have an important advantage: in constructive mathematics, a proof of an existential propositions is always carried out by giving an effective method of construction of a *witness*; i.e. an algorithm. This algorithm will be provided together with its proof of correctness. In summary, a formalization of constructive projective geometry can provide the synthesis of correct algorithms to solve problems in the areas of application of this discipline.

On the other hand, a formalization of projective geometry in Agda may be useful to validate the constructivity of certain proofs given in the literature. Since Agda is based on intuitionistic type theory, we can conclude that these proofs are built in accordance with the constructive method.

Outline. The paper is organized as follows. In Section 2, we present the formalization of the apartness relation and setoids. Section 3 describes the representation of the projective plane and its axioms in Agda. In Section 4, a method is presented to prove equalities with expressions involving the *meet* and *join* functions of a projective plane. In Section 5, we summarize some proofs developed in our formalization, in particular we comment about the implementation of the principle of duality. In Section 6, an overview of the implementation of Fano and Desargues properties is presented. In Section 7, we describe the formalization of harmonic conjugates and the proofs of existence and uniqueness thereof. Finally, conclusions and related work

[2] All the code described in this paper is freely available from the git repository: https://github.com/GuillermoCalderon/ProjectiveGeometryInAgda

are discussed in Section 8.

2 Apartness and Constructive Setoids

In the constructive formulation of projective geometry that we are following, it is necessary to define the equality relation as derived from a primitive *apartness* relation (i.e inequality). A detailed explanation of this methodology can be found in [24] and [15]. In this section we describe our implementation of *apartness* and *setoids*.

An *apartness relation* on a set A is a binary relation $\sharp$ on A satisfying three properties: irreflexivity, symmetry [3] and cotransitivity. It is implemented in Agda as a *dependent record* which takes A and $\sharp$ as parameters and whose fields postulate the three required properties:

```
record IsApartness
  {a b}{A : Set a}
  (_♯_ : A → A → Set b) : Set (a ⊔ b) where
  field
    irreflexive    : ∀ {x}     → ¬(x ♯ x)
    symmetry       : ∀ {x y}   → x ♯ y → y ♯ x
    cotransitivity : ∀ {x y} z → x ♯ y → z ♯ x ⊎ z ♯ y
```

In order to obtain a definition as general as possible, we consider that A belongs to the universe Set_a for a generic level a. Analogous assumptions are adopted for all objects in our formalization. We will not give details about levels in the rest of this paper. The reader unfamiliar with *universe polymorphism* [20] in Agda can just ignore them without loss of understanding. Note that some arguments are declared as implicit (the ones enclosed between curly brackets). Implicit arguments have a nice property: in most cases we can omit these arguments and Agda will try to infer them from the context. We use implicit arguments very often in our implementation. We will not explain why some arguments are declared as implicit and others are not. In general, it is decided by some sort of heuristics given by practice.

Equality. We define the equality relation $\approx$ as the negation of apartness: $x \approx y \equiv \neg(x \sharp y)$. This approach is slightly different from the one adopted in most axiomatizations of projective geometry ([7,15,24]) where the equality is a primitive concept and the relationship between equality and apartness is given by an additional axiom (*tightness*): *if* $\neg(x \sharp y)$ *then* $x = y$. The converse of tightness follows from irreflexivity. Thus, a logical equivalence is established among equality and the negation of apartness. By simplicity, we obtain this equivalence defining the equality relation as an alias of *not-apartness* (without affecting the system as a whole) As expected, $\approx$ is proved to be an *equivalence* relation.

Setoids. A setoid is a pair $\langle A, \sharp \rangle$ where A is a set and $\sharp$ an apartness relation on

[3] In fact, *symmetry* it is not required because we can derive it from the other two properties. However, symmetry is usually included in the definition of *apartness*

A[4]:

```
record Setoid♯ a b : Set (suc (a ⊔ b)) where
  field
    {Carrier}  : Set a
    _♯_        : Carrier → Carrier → Set b
    isApartness : IsApartness _♯_
```

Relations on Setoid. In order to work with setoids, we need to lift some set based operations on setoids. This process consists in the definition of a record which contains a field representing the operation at the level of the carriers together with another field that asserts the compatibility of the operation with the apartness/equality relation. In the following code, $\langle \_ \rangle$ denotes an operator which returns the carrier of a setoid.

```
record IsRel♯ {a₁ b₁ a₂ b₂ c}
  (S₁ : Setoid♯ a₁ b₁)(S₂ : Setoid♯ a₂ b₂)
  (R : ⟨ S₁ ⟩ → ⟨ S₂ ⟩ → Set c) :
  Set (a₁ ⊔ b₁ ⊔ a₂ ⊔ b₂ ⊔ c) where
  field
    sound : ∀ {a b c d}
            → Setoid♯._≈_ S₁ a b
            → Setoid♯._≈_ S₂ c d
            → R a c → R b d
record Rel♯ {a₁ b₁ a₂ b₂} c
        (S₁ : Setoid♯ a₁ b₁)(S₂ : Setoid♯ a₂ b₂)
  : Set (suc (a₁ ⊔ b₁ ⊔ a₂ ⊔ b₂ ⊔ c)) where
  field
    R     : ⟨ S₁ ⟩ → ⟨ S₂ ⟩ → Set c
    isRel : IsRel♯ S₁ S₂ R
```

Equality and Rewriting. The field `sound` of binary relations on setoids, gives place to a method of constructing proofs by substitution of equals for equals preserving the relation (also known as *rewriting*). If $\_ \bowtie \_$ is a binary relation between setoids and we have a proof of $a_1 \bowtie b_1$ and proofs of $a_1 \approx a_2$ and $b_1 \approx b_2$ then we obtain a proof of $a_2 \bowtie b_2$.

We introduce a couple of operators which allow us to mimic a very common pattern used in informal mathematical notation. For instance, the expression $P = Q \in A = B$ stands for a proof that $P \in B$, provided we have proofs of $P = Q$, $A = B$ and $Q \in A$.

[4] We use the name `Setoid♯` standing for an apartness based setoid. The identifier `Setoid` is preserved to denote a classical setoid such as it is defined in Agda standard library.

We implement two operators for rewriting which apply substitution over the left ($\langle\_\rangle\Leftarrow\_$) and right ($\_\Rightarrow\langle\_\rangle$) arguments of the relation as follows

```
⟨_⟩⇐_          : ∀ {a₁ a₂ b} → a₂ ≈ a₁ → R a₁ b → R a₂ b
⟨ a₂≈a₁ ⟩⇐ Ra₁b = sound (sym a₂≈a₁) refl Ra₁b
_⇒⟨_⟩          : ∀ {a₁ a₂ b} → R b a₁ → a₁ ≈ a₂ → R b a₂
Rba₁ ⇒⟨ a₁≈a₂ ⟩ = sound refl a₁≈a₂ Rba₁
```

Overloading. We need to work with several setoids at the same time. In order to overload the apartness operators for different setoids, we use an Agda feature called *instance arguments* [5]. The idea is rather simple: we can open the module `Setoid♯` with a special directive: `open Setoid♯ ⦃...⦄`. This directive allows us to access all the operations of the module `Setoid♯` and we can omit the particular instance argument which will be inferred by Agda from the context by a special *instance resolution algorithm* [5].

3 Representing the Projective Plane

In this section, we describe the implementation in Agda of the system of axioms of projective geometry according to [15].

Point, Lines and Basic Relations. The basic objects of projective plane geometry are *points* and *lines* which constitute setoids; i.e. they are equipped with an apartness relation. In addition, we have the incidence relation ($\in$) and the outside relation ($\notin$). Both relations are assumed to be compatible with equality.

We start defining a record `IsProjectivePlane` which constitutes the main structure of our formalization of projective geometry.

```
record IsProjectivePlane {a b c d e}
  { Point♯ : Setoid♯ a b }
  { Line♯ : Setoid♯ c d }
  (Incidence Outside : Rel♯ e Point♯ Line♯)
  : Set (a ⊔ b ⊔ c ⊔ d ⊔ e) where
```

The names *Incidence* and *Outside* will be soon forgotten since we will use the raw relations $\in$ and $\notin$ at the level of the carriers. The first step is to give some convenient definitions that allows us to access the components of the structure.

```
Point = ⟨ Point♯ ⟩
Line = ⟨ Line♯ ⟩
_∈_ : Point → Line → Set e
_∈_ = Rel♯.R Incidence
_∉_ : Point → Line → Set e
_∉_ = Rel♯.R Outside
```

[5] Instance arguments of Agda can be seen as an equivalent of Haskell type classes.

Outside Relation. In [15] and [7] the *outside* relation is defined positively (i.e. without negation) from the incidence relation and point apartness: $P \notin l \equiv (\forall Q)(Q \in l) \to P \sharp Q$. We take a different approach in our formalization: we do not give an explicit definition of the relation outside. Instead, we add an axiom (C0) asserting that $P \notin l \Rightarrow (\forall Q)(Q \in l) \to P \sharp Q$. The converse of this implication is not postulated but it can be proved using the other axioms [6].

3.1 Axioms for Projective Plane

In this section we begin the description of the axioms of a projective plane which are represented as fields of the record `IsProjectivePlane`.

Axiom C0 was explained in the previous section:

$$\mathsf{C0} : \forall\, \{P\ l\} \to P \notin l \to (\forall\, \{Q\} \to Q \in l \to P \sharp Q)$$

A first group of axioms establishes the existence of a line and of a point external to it (axiom C1 of Mandelkern). These axioms exclude the possibility of trivial cases of projective plane

$$\begin{aligned} \mathsf{P_0} &: \mathsf{Point} \\ \mathsf{l_0} &: \mathsf{Line} \\ \mathsf{P_0 \notin l_0} &: \mathsf{P_0} \notin \mathsf{l_0} \end{aligned}$$

The next group of axioms (C2 and C3) expresses conditions of existence and uniqueness of the line passing by two distinct points as well as its dual (a point on two distinct lines).

Here, the relation of apartness becomes essential to ensure the existence of a line passing by two distinct points.

$$\begin{aligned} \mathsf{join} &: \forall\, \{P\ Q : \mathsf{Point}\} \to P \sharp Q \to \mathsf{Line} \\ \mathsf{join_1} &: \forall\, \{P\ Q\}\ \{P\sharp Q : P \sharp Q\} \to P \in \mathsf{join}\ P\sharp Q \\ \mathsf{join_r} &: \forall\, \{P\ Q\}\ \{P\sharp Q : P \sharp Q\} \to Q \in \mathsf{join}\ P\sharp Q \\ \mathsf{unq\text{-}join} &: \forall\, \{P\ Q\}\ (P\sharp Q : P \sharp Q) \\ &\to \forall\, \{l\} \to P \in l \to Q \in l \to l \approx \mathsf{join}\ P\sharp Q \end{aligned}$$

A function join is given, which receives two points and a proof of their apartness, and returns a line (existence). The functions $\mathsf{join_1}$ and $\mathsf{join_r}$ allow to prove that the join of two points indeed passes through them. Finally, the function unq-join asserts that all lines passing thorough two points are equal to the join of these points.

Note the difference with the traditional notation PQ to represent the line passing by the points P and Q. In our implementation this line is denoted by the expression $(\mathsf{join}\ P\sharp Q)$, where $P\sharp Q$ is a proof that P and Q are distinct.

[6] However, the converse is never used in our implementation.

We define analogously the functions meet, meet_1, meet_r and unq-meet which represent the dual part of the family of *join* operations.

A few axioms are formulated in order to rule out too simple models of projective plane (C4). Then the existence of three distinct points on any line is assumed:

$$
\begin{aligned}
&\text{point}_1\ \text{point}_2\ \text{point}_3 : \text{Line} \to \text{Point}\\
&\in\text{-point}_1 : \forall\ l \to \text{point}_1\ l \in l\\
&\in\text{-point}_2 : \forall\ l \to \text{point}_2\ l \in l\\
&\in\text{-point}_3 : \forall\ l \to \text{point}_3\ l \in l\\
&\text{point-i}\sharp\text{j} : \forall\ l \to \text{point}_1\ l \sharp \text{point}_2\ l\\
&\qquad\qquad \times\ \text{point}_1\ l \sharp \text{point}_3\ l\\
&\qquad\qquad \times\ \text{point}_2\ l \sharp \text{point}_3\ l
\end{aligned}
$$

A lot of propositions could be inferred from the axioms above if we were working using classical reasoning. Several of these conclusions are not constructively valid. Therefore, it is necessary to add some other axioms in order to obtain a useful system and at the same time, preserving its constructive foundation. We keep Mandelkern's denomination for these axioms.

Axiom C5 permits to infer that two lines are distinct if a point belongs to one of the lines and is outside the other. In turn, C6 establishes a strong relation between $\notin$ and $\in$. The last axiom C7 provides another view about the uniqueness of the intersection of two lines, but involving $\notin$ instead of $\in$. It has the particularity that its conclusion is a disjunction

$$
\begin{aligned}
&\text{C5} : \forall\ \{l\ m\ P\} \to P \in l \to P \notin m \to l \sharp m\\
&\text{C6} : \forall\ \{P\ l\} \to \neg\ (P \notin l) \to P \in l\\
&\text{C7} : \forall\ \{l\ m\ P\} \to (l\sharp m : l \sharp m) \to P \sharp \text{meet}\ l\sharp m \to P \notin l \uplus P \notin m
\end{aligned}
$$

The converse of C6 is easily proved.

3.2 Comparing with other Axiom Systems

Since the seminal work of Heyting [7], several different systems were given to define projective geometry in an acceptable constructive way. Here, we investigate the relationship between our formalization (mostly based on [15] which in turn is similar to Heyting's formulation) with the ones given by von Plato [24] and van Dalen [22].

Von Plato system defines a core for all the geometries which is called *apartness geometry*. In this approach all the usual negative relations are considered as primitive and positive. Then, the relations of apartness (for points and lines) and the outside relations are considered as given and the relations of equality and incidence are defined as the respective negations of these primitive relations. The collection of axioms are very simple and symmetric. Several of them have disjunctive conclusions.

We give a formal proof that every projective plane as we define in Section 3 constitutes an *apartness projective geometry* in the sense of von Plato. The converse requires to add to von Plato system some axioms postulating non-degeneracy conditions (like axioms C1 and C4 of Mandelkern). With this addition, we formalize

the correspondence of a projective geometry à la von Plato with a Mandelkern's projective plane.

On the other hand, van Dalen describes an axiom system for projective geometry where the only primitive relation is the outside relation ($\notin$). All the other binary relations of the system are defined in function of the referred outside relation. We implement the proof of equivalence of the van Dalen's system with ours.

We consider these equivalences very important and useful, because they allow us to work freely with different collections of axioms. Our implementation of each system of axioms is represented as a dependent record in Agda. Thus, we can make available any of the systems by importing the respective module.

4 Equational Reasoning with *join* and *meet*

The functions join and meet permit the construction of complex expressions representing points. Often, we find different expressions (compositions of join and meet) denoting the same object (point or line). The cause of this fact relies on the uniqueness properties expressed by the functions unq-join and unq-meet. Below, we present some algebraic rules expressing equalities between expressions constructed with join and meet. We use a simplified notation: AB_q denotes an expression of the form (join A B q) where q is a proof of $A \sharp B$. The sub index q is omitted when it is irrelevant.

$$\frac{}{AB_p \approx AB_q}\ {\approx}\text{join} \qquad \frac{}{AB \approx BA}\ \text{join-comm}$$

$$\frac{A_1 \approx A_2 \quad B_1 \approx B_2}{A_1B_1 \approx A_2B_2}\ \text{join-cong} \qquad \frac{A_1 \approx A_2 \quad B_1 \approx B_2}{A_1B_1 \approx B_2A_2}\ \text{join-flip-cong}$$

Fig. 1. Rules for equational reasoning

We have dual rules for meet. We implement simple functions which allow to prove these algebraic equations in a rather direct manner. An interesting feature of this implementation is that all the arguments of these functions are implicit. Thus, we can combine the functions in a very direct way, guided by the simple intuition of the rules.

In addition, these operators can work together with the operators of the library EqReasoning of Agda ([18], [1]) providing a nice syntax to write some proofs of equality.

Playing with the system, one soon discovers that most proofs follow the pattern of establishing a relation among certain geometric objects (any of the basic relations, apartness, equality, incidence and outside). Thus, the axioms required are invoked, and usually we need to apply rewriting in order to obtain the relation involving the correct objects. Overloaded operators for rewriting and the combinators for equality described in Section 4 are extensively used in the construction of proofs in our system.

5 Propositions and Duality

Once we have implemented the previous modules with the definitions of projective planes and the libraries for setoids and equational reasoning, we can construct the proofs of propositions such as the ones appearing in Section 2 of [15] and most traditional books.

We provide also a formal proof of the *principle of duality* considered one of the most typical results of projective geometry. The classical formulation of this principle expresses that: *For every valid proposition, it is also valid the dual proposition which is obtained from the first one swapping the words "line" by "point", "pass" by "lie" and so on.* This informal statement seems rather a *meta-theorem* than a geometric proposition. However, it can be formalized in a very succinct way like a function among projective planes.

```
duality : ∀  {a b c d e} {Point♯ : Setoid♯ a b} {Line♯ : Setoid♯ c d}
             {∈ ∉ : Rel♯ e Point♯ Line♯}
         → IsProjectivePlane ∈ ∉
         → IsProjectivePlane (flip♯ ∈) (flip♯ ∉)
```

The function flip♯ interchanges the order of the arguments of a binary relation on setoids. The proof of the principle of duality indicates how to construct a dual projective plane from a given projective plane.

6 Fano and Desargues

In planar projective geometry, the classic theorems of Fano and Desargues can not be proved. Therefore, we assume them as new axioms. More precisely, we define extensions of the previously defined projective plane by adding fields representing these axioms.

Fano axiom is about *quadrangles*, and Desargues has to do with *triangles*. Quadrangles and triangles are instances of a kind of figure known as *complete n-point*. We define these concepts in our implementation providing modules with the necessary tools to work comfortably with them.

We define a structure FanoProjectivePlane as an extension of the structure ProjectivePlane by the addition of a new field that represents the Fano axiom. In turn, another structure Desarguesian is obtained as the extension of FanoProjectivePlane with a field corresponding to Desargues axiom. The principle of duality that was proved for the projective plane is generalized for the new defined structures by giving the proofs for the dual versions of Fano an Desargues axioms. Although conceptually simple, these proofs are rather extensive and verbose (about a thousand of lines of Agda code). Some interesting mathematical machinery is implemented in these modules in order to work appropriately with the symmetry and circularity of the kind of figures involved. We do not provide a detailed explanation to this part of our implementation which can be consulted in the repository by the interested reader.

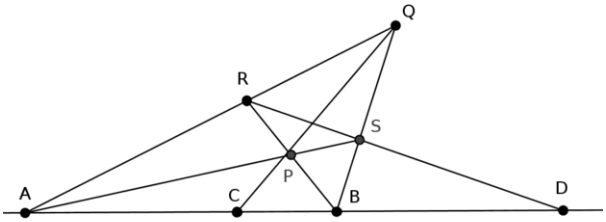

Fig. 2. A harmonic configuration for C with respect to A and B

7 Harmonic Conjugates

Harmonic conjugates can be defined in Euclidean geometry in terms of metric concepts (the cross ratio). von Staudt [25] was the first one to propose a synthetic definition based in quadrangles. We follow the definition given in [15] which better accomplishes the required constructivity since the definition is uniform over all the points of the base line.

7.1 Definition of Harmonic Conjugate

Assume two distinct points A and B. Let C be any point lying on the line AB. A *harmonic configuration* of C with respect to A and B is determined by a line l passing by C and different to AB, and a point R outside l and AB.

From the line l and the point R we can construct a point D which is called the harmonic conjugate of C with respect to A, B. The method to obtain the conjugate of C proceeds as follows: Let Q be the intersection of l and AR. Let P be the intersection of l and BR. Let S be the intersection of AP and BQ Then, the *harmonic conjugate* of C with respect to A and B (relative to the configuration (l, R)) is the point $D = RS \cap AB$.

In order to have a valid definition, we must show that all the steps are well defined, i.e. all joins and meets are applied over different points and lines. In our formalization, this is forced by the Agda type checker since the functions join and meet require as argument a proof of apartness of the objects involved. For instance, to refer to the line AR we need a proof of $A \sharp R$ and to consider the intersection of l and AR we need a proof of $l \sharp AR$ and so on.

Harmonic Configuration. Given A and B distinct points and C a point belonging to AB, a harmonic configuration for C is the collection of lines and points constructed (following the method described above) from a particular choice of l and R in order to obtain the point D, (the harmonic conjugate of C with respect to A and B).

We represent a harmonic configuration as a record:

```
record HarmonicConf
  {A B} {A♯B : A ♯ B} {C}
  (C∈AB : C ∈ join A♯B) : Set (a ⊔ b ⊔ c ⊔ d ⊔ e) where
```

```
field
  l       : Line
  C∈l     : C ∈ l
  l♯AB    : l ♯ join A♯B
  R       : Point
  R∉l     : R ∉ l
  R∉AB    : R ∉ join A♯B
```

The code above shows the fields of the record `HarmonicConf`, namely the line l and the point R together with the conditions they have to fulfill. In the same module `HarmonicConf` we define the points and lines required to construct the harmonic conjugate as well as the necessary proofs of apartness for each *join* and *meet* invoked. In particular, the points of the quadrangle (see below) are defined this way:

```
P Q S D : Point
P = meet RB♯l
Q = meet RA♯l
S = meet AP♯BQ
D = meet RS♯AB
```

Representing the Harmonic Conjugate. According to the previous development, the harmonic conjugate is defined as a function which returns a point from a given configuration:

```
HConjugate      : ∀ {A B C : Point}{A♯B : A ♯ B}{C∈AB : C ∈ join A♯B}
                  → HarmonicConf C∈AB → Point
HConjugate HC = HarmonicConf.D HC
```

Note the only explicit argument is the structure (record) corresponding to the harmonic configuration. Inside this record we give the construction of the point D which is the field selected for the configuration. All the other arguments are implicit, and they will be inferred from the argument `HC`.

Quadrangles and Harmonic Conjugate. It is usual to define the harmonic conjugate by using a quadrangle. In such definition, a quadrangle $PQRS$ is constructed in such a way that the lines QR and PS meet in A, the lines PR and QS meet in B, and the line PQ pass by C. Then, the harmonic conjugate of C with respect to A and B is the intersection of the lines RS and AB. This construction, is only possible when C does not coincide with A nor B. For this reason, we follow the definition given in [15] where we do not need to determine whether C is equal to A or B in order to define the harmonic conjugate.

7.2 Existence of the Harmonic Conjugate

The existence of the harmonic conjugate is in general admitted without any proof in the standard literature. In accordance with our constructive approach we have to

provide a method to construct at least one harmonic configuration from any given points A, B and C.

$$\begin{aligned}\exists\mathsf{HConf} : \forall\ & \{A\ B\ C : \mathsf{Point}\}\{A\sharp B : A \sharp B\}\\ & \to (C{\in}AB : C \in \mathsf{join}\ A\sharp B)\\ & \to \mathsf{HarmonicConf}\ C{\in}AB\end{aligned}$$

By axiom C4 dual, we can obtain two distinct lines passing by C. By cotransitivity, at least one of these lines is distinct from AB. Then, we have the required line l passing by C and distinct from AB. The existence of the point R, outside both AB and l, is harder to prove. It follows also by an appropriate composition of axiom C4 and cotransitivity.

7.3 Uniqueness of Harmonic Conjugate

From the definition above, the conjugate of a given point with respect to A and B, will depend on the line l and the point R chosen (i.e. the configuration considered). One of the main results in projective geometry is that the harmonic conjugate does not depend on the configuration chosen. In other words, if we consider two configurations HC and HC′ the conjugates D and D' obtained using respectively HC and HC' are the same.

$$\begin{aligned}\mathsf{uniqueness} : & (A\ B\ C : \mathsf{Point})\\ & \to (A\sharp B\quad : A \sharp B)\\ & \to (C{\in}AB : C \in \mathsf{join}\ A\sharp B)\\ & \to (HC\ HC' : \mathsf{HarmonicConf}\ C{\in}AB)\\ & \to \mathsf{HConjugate}\ HC \approx \mathsf{HConjugate}\ HC'\end{aligned}$$

The construction of this proof is the main goal of this paper. We are going to explain the general scheme used in the construction of this formal proof and to illustrate some parts with more detail.

7.3.1 Constructive Proofs by Cases.

It is very common, in classic mathematics, to construct a proof by cases which derive from certain variant of the *Law of Excluded Middle* (LEM). If we prove a proposition ϕ from the assumption ψ and we also prove ϕ from assumption $\neg\psi$, then we have a proof ϕ (under no assumptions). This method is not valid if we are working constructively and do not accept LEM as a general valid principle. However, in the particular case where we want to prove a contradiction ($\bot$) we can proceed as follows: From the assumption ϕ, to prove a contradiction, and from the assumption $\neg\phi$, to prove a contradiction. Then we have a proof of $\bot$ (without assumptions). The first contradiction gives a proof of $\neg\phi$ and cancels the assumption ϕ. The second contradiction takes this proof of $\neg\phi$ and gives a proof of the absurdity ($\bot$).

This method can be generalized to an arbitrary number k of premises which are canceled by contradiction in sequence. Let $\phi_1, \ldots, \phi_k$ be a collection of assumptions.

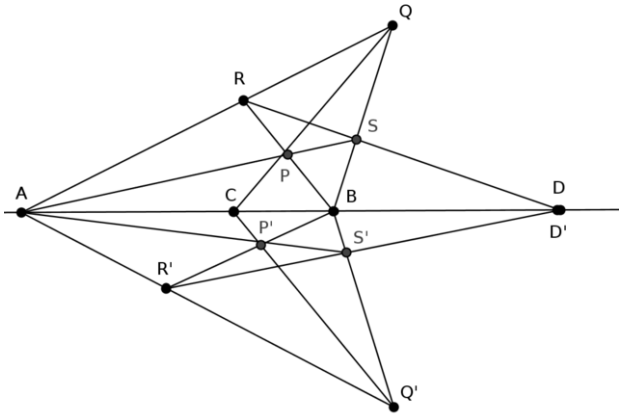

Fig. 3. Two harmonic configurations. The case of completely disjoint quadrangles

We have to provide $(k + 1)$ proofs of $\bot$ in sequence. In each step we cancel a premise by negation. Finally, we obtain a proof of $\bot$ independent of the assumptions $\phi_1, \ldots, \phi_k$.

In the next section, we explain how to apply this pattern of reasoning to the construction of a proof for uniqueness of the harmonic conjugate.

7.3.2 The Proof of Uniqueness

In this section, we explain how the proof of uniqueness was constructed and formalized in Agda. This is a rather extensive proof which consideres a lot of different cases in order to cover all the possibilities [7] .

The proof takes as input the points A, B and C as defined above and two harmonic configurations HC and HC'. Let D and D' be the harmonic conjugates with respect to HC and HC' respectively. We want to construct a proof of $D \approx D'$ which is defined as the negation of $D \sharp D'$. In other words, a proof of $D \approx D'$ is obtained by assuming $D \sharp D'$ and deriving a contradiction from this assumption.

In addition to the global assumption of $D \sharp D'$ we incorporate several additional assumptions which allow us to prove a contradiction by cases as explained in the previous section. These assumptions have to do with how distinct the configurations HC and HC' are.

For instance, we will describe the proof for the case where the two configurations are completely disjoint.

We need eleven assumptions in order to configure this case:

$C \sharp A$, $C \sharp B$, $l \sharp l'$, $R \sharp R'$, $R' \notin RB$, $R' \notin RA$, $SP \sharp S'P'$, $SQ \sharp S'Q'$, $O \notin RS$, $O \notin R'S'$, $S \sharp S'$.

Note that all the premises are cases of the apartness $(\_\sharp\_)$ or the outside $(\_ \notin \_)$ relation. This is important, since from the negation of $A \sharp B$ we can infer $A \approx B$ (by definition) and the negation of $A \notin r$ entails that $A \in r$ (by axiom C6).

[7] The complete code of the proof comprises about 5000 lines and it is distributed among about 15 files.

The first and second assumptions ensure the existence of the quadrangles for both configurations. The condition $l \sharp l'$ permits to deduce that $P \sharp P'$ and $Q \sharp Q'$. From $R' \notin RB$, and $R' \notin RA$ we can prove that the triangles PQR and $P'Q'R'$ are perspective from the axis AB. By Desargues converse, PQR and $P'Q'R'$ are perspective from a center. Let be $O = PP' \cap QQ'$ this center of perspectivity.

The triangles PQS and $P'Q'S'$ are also perspective from the axis AB (using the assumptions $SP \sharp S'P'$, and $SQ \sharp S'Q'$). By Desargues converse, these triangles are perspective by a center. It is immediate to see that this center is equal to O. We can conclude that the triangles PRS and $P'R'S'$ are perspective from the center O (the assumptions $O \notin RS$, $O \notin R'S'$ and $S \sharp S'$ are required) Applying now Desargues, these triangles are also perspective from an axis. We can easily determine that this axis is the line AB and hence $D \approx D' = RS \cap R'S'$. From the general assumption $D \sharp D'$, we obtain the contradiction for this case.

Just this case is usually presented in most books of projective geometry as a complete proof of uniqueness of the harmonic conjugate. Moreover, some of the required premises (like those involving the point O) are omitted.

Veblen and Young [23] present a proof like the one above described. They argue that for the case where a vertex or side of HC coincides with a vertex or side of HC$'$, we can consider a third configuration HC$''$ whose vertices and sides are distinct from those of HC and HC$'$. Hence, we can apply the previous proof to the configurations HC and HC$''$ and then to the configurations HC$'$ and HC$''$. By transitivity, the equality of D and D' is deduced. If we want to formalize this argument, we have to give a method to construct this third configuration. Moreover, we should provide a method for each of the several cases where one of the previous hypothesis fails. This task seems to be rather complex in the context of a constructive approach [8] .

In turn, Coxeter's book [4] presents only the proof for the case of disjoint configurations and the proof for the other cases are left to the reader.

With respect to constructive approaches, Heyting [7] gives a quadrangle based definition of harmonic conjugates in the context of projective geometry of the space, providing a construction which only applies to points different from the base points (A and B). At the same time, the proof given by Mandelkern [15], is not rigorous enough in the construction of the sequence of contradiction and it has some inaccuracies [16].

In addition to the proof based upon the eleven assumptions, in our implementation, we have considered a lot of more cases negating the assumptions in sequence one at a time. For each case, we have to obtain a contradiction. The contradiction for some cases is rather evident, whereas for others a considerable effort is required to deduce it. Mostly, we obtain the required result by a number of applications of Desargues just as we do for the case of disjoint configurations. For a number of cases it is possible to take advantage of the symmetry of the configurations and to reuse the proof already given for an analogous situation.

[8] Note the required method to obtain a third configuration becomes trivial if we consider a plane embedded in a projective space.

8 Conclusion and Related Work

We have described a formalization of a basic fragment of constructive projective plane geometry which covers: the representation of basic objects and relations, the definition of projective plane with its axioms, basic propositions of incidence, the principle of duality, the definition of complete n-point, representation of Fano and Desargues axioms and proofs of their dual formulations, the definition of the harmonic conjugate and proofs of its existence and uniqueness. The implementation has been restricted to use only valid methods from constructive mathematics. We implement the axiom system described in [15] with some minor variants. In addition, the equivalence with other well-known constructive axiom systems was proved.

The proof of uniqueness of the harmonic conjugate turns out to be a quite complex proof when carried out just by using constructively valid methods. In fact, we could not find a complete and rigorous presentation of this proof in the literature. Therefore, a formal and machine-checked construction of such a proof constitutes a major contribution of this paper.

We have based our development on the programming language features of Agda rather than on the proof-assistant ones. In particular, we have not constructed proofs by tactics. We have taken advantage of a number of important features of the language. Modules and dependent record types have played an important role allowing us to implement setoids as an *abstract data type* and to view lines and points as instances of this ADT. In addition, the mechanism of *implicit instances* has provided the ability for overloading the operators of this ADT. We almost did not need any additional data type: only natural numbers and finitary sets are used in order to provide the definition of complete n-point. With respect to the logic involved in the implementation: we have used the standard connectives of first order intuitionistic logic, namely implication, negation, disjunction, conjunction and quantifiers.

Induction and recursion have been rarely used along our implementation. This is rather unusual for a formalization in type theory, but it is reasonable in a case where the main objects are ADTs instead of inductively defined data types. Accordingly, pattern matching is scarcely used (only to process disjunctions and finitary sets).

One of the goals of our work has been the implementation of a framework where one can easily construct proofs of projective geometry without using automatic tactics. By "easily constructing" we mean to write formal proofs in a similar way as it is done in mathematics textbooks. In other words, we want to reduce the complexity of a formal proof compared with a paper proof. We have obtained very elegant and simple proofs combining the operators provided by our implementation (e.g. rewriting, join-meet rules, eq-reasoning, circularity of configurations). However, some proofs (namely: Desargues an Fano duals and unicity of harmonic conjugates) are excessively verbose compared with their textbook versions. Some additional work is required in order to reduce this complexity. We conjecture that an appropriate redesign of some of the basic modules of our implementation would help to reach this goal. Moreover, some kind of automatization would be useful to avoid boilerplate

code (e.g. proofs of equality for lines and points).

Related Work. To the best of our knowledge, there are no other formalizations of geometry (of any kind) implemented in Agda.

There exists an early formalization written in Alf (an ancestor of Agda) [13] by von Plato [24]. This is a very simple formalization of von Plato axiom system. A more advanced implementation is [8] which formalizes von Plato constructive geometry in Coq. These works only present a small fragment of projective geometry without covering advanced concepts such as Desargues theorem and harmonic conjugates. However, they use the same representation for line/point existence and uniqueness: with functions *join* and *meet* defined as in our implementation. The definition of apartness relation is also very similar.

If we restrict ourselves to projective geometry formalizations in type theory, we should mention the work by Magaud, Narboux and Schreck [11,12]. They describe Coq implementations of the basic concepts of synthetic projective geometry using a tactic-based method to construct the proofs. They cover some complex theorems like Desargues and the principle of duality. There are some points in common with our formalization, namely the representation of the projective plane as a record and the proof of duality carried out by providing a function among records. The main and capital difference is that their formalization does not follow a pure constructive approach since the incidence and equality relations are postulated to be decidable.

At last, we note that we are not aware of any formalization of projective geometry covering the area of harmonic conjugates.

Future Work. This work is part of a more ambitious project consisting in a complete formalization of constructive projective geometry. Future topics to cover will be: projectivities and conics. In addition, there are a number of interesting related problems to be investigated such as: the relationship of projective geometry with affine geometry, i.e. the extension problem ([21], [14]); implementation of the coordinatization of the projective plane; relation with algebraic models of projective geometry like *Grassman-Cayley* algebra, geometric algebra, etc. [10].

Acknowledgment

I would like to thank Alberto Pardo and the anonymous referees for their useful comments. I also thank M. Mandelkern for helpful comments about his work on constructive projective geometry.

References

[1] Agda developers, *Agda standard library*, http://wiki.portal.chalmers.se/agda/pmwiki.php/Libraries/StandardLibrary, [Date Accesed: 2017-04-05].

[2] Beutelspacher, A. and U. Rosenbaum, "Projective geometry - from foundations to applications." Cambridge University Press, 1998, I-X, 1-258 pp.

[3] Birchfield, S., *An Introduction to Projective Geometry (for computer vision)*, http://robotics.stanford.edu/~birch/projective/ (1998), Date accessed: 2017.

[4] Coxeter, H. S. M., "Projective geometry," Springer Science & Business Media, New York, 2003.

[5] Devriese, D. and F. Piessens, *On the bright side of type classes: instance arguments in Agda*, in: *Proceeding of the 16th ACM SIGPLAN international conference on Functional Programming, ICFP 2011, Tokyo, Japan, September 19-21, 2011*, 2011, pp. 143–155.

[6] Herman, I., "The use of projective geometry in computer graphics," Springer-Verlag Berlin ; New York, 1992, viii, 146 p. : pp.

[7] Heyting, A., *Zur intuitionistischen axiomatik der projektiven geometrie*, Mathematische Annalen **98** (1928), pp. 491–538.

[8] Kahn, G., *Constructive geometry according to Jan von Plato*, Coq contribution. Coq **5** (1995), p. 10.

[9] Klein, F., "Das Erlanger Programm," 253, Akademische Verlagsgesellschaft Geest & Portig, 1974.

[10] Li, H., "Invariant algebras and geometric reasoning," World Scientific, Singapore, 2008.

[11] Magaud, N., J. Narboux and P. Schreck, *Formalizing projective plane geometry in Coq*, in: *Automated Deduction in Geometry - 7th International Workshop, ADG 2008, Shanghai, China, September 22-24, 2008. Revised Papers*, 2008, pp. 141–162.

[12] Magaud, N., J. Narboux and P. Schreck, *A case study in formalizing projective geometry in Coq: Desargues theorem*, Comput. Geom. **45** (2012), pp. 406–424.

[13] Magnusson, L. and B. Nordström, *The ALF proof editor and its proof engine*, in: *Types for Proofs and Programs, International Workshop TYPES'93, Nijmegen, The Netherlands, May 24-28, 1993, Selected Papers*, 1993, pp. 213–237.

[14] Mandelkern, M., *Constructive projective extension of an incidence plane*, Transactions of the American Mathematical Society **366** (2014), pp. 691–706.

[15] Mandelkern, M., *A constructive real projective plane*, Journal of Geometry (2015), pp. 1–42.

[16] Mandelkern, M., *About the proof of uniqueness of harmonic conjugates*, Personal Communication (by email) (2017).

[17] Nordström, B., K. Petersson and J. Smith, "Programming in Martin-Löf's type theory: an introduction," International series of monographs on computer science, Clarendon Press, 1990.

[18] Norell, U., "Towards a practical programming language based on dependent type theory," Ph.D. thesis, Department of Computer Science and Engineering, Chalmers University of Technology, SE-412 96 Göteborg, Sweden (2007).

[19] Norell, U., *Dependently typed programming in Agda*, in: *Proceedings of TLDI'09: 2009 ACM SIGPLAN International Workshop on Types in Languages Design and Implementation, Savannah, GA, USA, January 24, 2009*, 2009, pp. 1–2.

[20] The Agda Team, *Universe polymorphism. The Agda wiki*, http://wiki.portal.chalmers.se/agda/pmwiki.php/ReferenceManual/UniversePolymorphism, [Date Accesed: 2017-07-03].

[21] van Dalen, D., *Extension problems in intuitionistic plane projective geometry, I and II*, *Indagationes Mathematicae* , **66**, Elsevier, 1963, pp. 349–368.

[22] van Dalen, D., *'Outside' as a primitive notion in constructive projective geometry*, Geometriae Dedicata **60** (1996), pp. 107–111.

[23] Veblen, O. and J. W. Young, "Projective geometry," **2**, Ginn, 1918.

[24] von Plato, J., *The axioms of constructive geometry*, Annals of pure and applied logic **76** (1995), pp. 169–200.

[25] von Staudt, K. G. C., "Geometrie der Lage," Nürnberg, 1847.