Finding and Extracting Computational Content in Euclid's *Elements*

Ariel Kellison

Septemeber 2017

Outline

Introduction

Implementing Euclid's *Elements* and Computational Content

- Implementing a formalization of Euclid's *Elements* in Nuprl's Constructive Type Theory required that we focus on capturing and expressing computational content.
- The *computational content* of Euclid's geometry corresponds to *the geometric objects* that are *constructed by straightedge and compass*.

Why Implement SC constructions?

- The proofs are human readable and are fully checked; the programs they correspond to are provably correct.
- The geometry remains (in our implementation) *synthetic*. Instead of proofs that rely on numerical analysis, synthetic geometrical proofs retain the intuitive nature of geometric configurations and constructions.
- The constructions correspond to concepts in *constructive mathematics*; "Every theorem proved with [nonconstructive] methods presents a challenge: to find a constructive version, and to give it a constructive proof" -Errett Bishop.
- Pedagogical interest: computer science concepts can be taught in secondary school geometry classrooms.

Results and Future Work

- We have fully constructive proofs of the first 12 propositions of Euclid, including proposition 2 which has been considered "irreparably non-constructive."
- Our formalization of Euclid's geometry allows us to approach modern problems in computer science, namely finding the convex hull of points. This is a result of out ability to reason synthetically about orientation.
- We have fully constructive proofs of *most* of Alfred Tarski's *Metamathematische Methoden in der Geometrie* , an influential modern system of Euclidean geometry.
- Future Work: connecting visual dynamic geometry tools to Nuprl, extending the concepts to secondary and undergraduate classrooms, extending the geometry (potentially to HoTT!), constructivizing the Coq library of proofs from *Metamathematische Methoden in der Geometrie* using the Nuprl versions, novel problems in RC constructions...

Where to begin?

Short History

There are other formalizations of geometry: Alfred Tarski, *Metamathematische Methoden in der Geometrie* in FOL and David Hilbert *Grundlagen der Geometrie* (not formal in the modern sense). *Metamathematische Methoden in der Geometrie* has been implemented in Coq.

Metamathematics

Tarski made geometry a topic of *metamathematical investigation* where information was obtained *not within, but about* the discipline.

Postulates, Axioms, and Propositions

Axiom (Hilbert)

I,*1. Two distinct points A and B always completely determine a straight line a.*

Postulate (Euclid)

Post.3 *To describe a circle with any centre and distance.*

Propositions as Types

Prop.1 *To construct an equilateral triangle on a given finite straight line.*

- In order to formalize Euclid's geometry we must then be able to simultaneously *construct* geometric objects and *prove* that the constructed objects have the properties required.
- If we interpret every statement that we might want to prove as a type, then constructing an element of that type proves the original statement.

Proofs as Programs

"Building proofs is a special case of building theoretical objects in general."

- Using Nuprl's Type Theory we are able to *extract* the construction, or *constructive content*, of the object, and utilize it as we would a program.

The properties of Euclid's objects

Primitive Type

There is a primitive type, Point.

Equality

$$a \equiv b \Leftrightarrow_{\text{def}} \neg(a \# b).$$

Our axioms then imply that this is an equivalence relation on points.

Primitive Relations

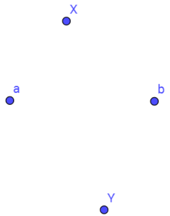
We take four primitive relations on points, $a \# b$, a left of bc, $ab \cong cd$, and a_b_c.

The Constructors

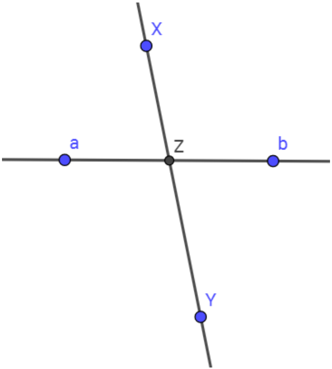
There are three construction axioms, corresponding to combinations of the basic constructions that can be done with the straightedge and *collapsing* compass.

- Straightedge-Straightedge
- Compass-Compass
- Straightedge-Compass

Straightedge-Straightedge (SS)



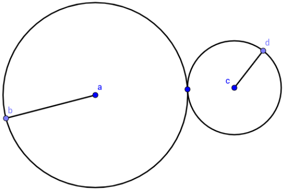
(a) given...



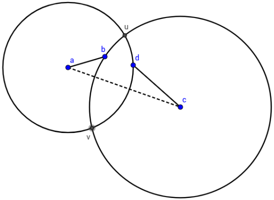
(b) ...SS constructs

Figure: Straightedge-Straightedge (SS) construction principle. Two points on opposite sides of a line will intersect the line at some point.

Compass-Compass



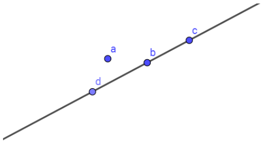
(a) 1a



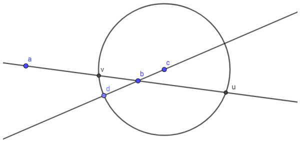
(b) 1b

Figure: Compass-Compass (CC) construction principle. The collapsing compass can be used to construct two circles and their potential points of intersection.

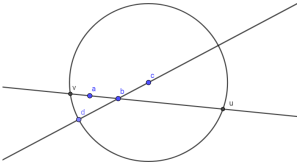
Straightedge-Compass



(a)



(b)



(c)

Figure: Straightedge-Compass (SC) construction principle. Use the straightedge first to determine a line segment, and then extend that segment using the collapsible compass.

Definitions used in the formalization of the constructors

- Not separated

$$a \equiv b \Leftrightarrow_{\text{def}} \neg(a \# b).$$

- Relative Orientation

$$a \text{ right of } bc \Leftrightarrow_{\text{def}} a \text{ left of } cb$$

- Point "line" separation

$$a \# bc \Leftrightarrow_{\text{def}} (a \text{ left of } bc \vee a \text{ right of } bc)$$

Definitions used in the formalization of the constructors

- Greater than

$$cd \geq ab \Leftrightarrow_{\text{def}} \neg\neg(\exists x.\ a\_b\_x \wedge ax \cong cd)$$

- Strictly greater than

$$cd > ab \Leftrightarrow_{\text{def}} (\exists x.\ a\_b\_x \wedge ax \cong cd \wedge b \# x)$$

- Collinearity

$$\text{Colinear}(a, b, c) \Leftrightarrow_{\text{def}} \neg(\neg(a\_b\_c) \wedge \neg(b\_c\_a) \wedge \neg(c\_a\_b))$$

Definitions used in the formalization of the constructors

Overlapping Circles

Circles $C(a,b)$ with center a and radius ab and $C(c,d)$ with center c and radius cd (strictly) overlap if there is a point p on $C(a,b)$ that is (strictly) inside $C(c,d)$ and a point q that is on $C(c,d)$ and (strictly) inside $C(a,b)$.

- $$\text{Overlap}(a,b,c,d) \Leftrightarrow \exists p,q.\ ab \cong ap \wedge cd \geq cp \wedge cd \cong cq \wedge ab \geq aq$$

- $$\text{StrictOverlap}(a,b,c,d) \Leftrightarrow$$
  $$\exists p,q.\ ab \cong ap \wedge cd > cp \wedge cd \cong cq \wedge ab > aq$$

Formalization of the constructors

- Straightedge-Straightedge:

$$(x \text{ left of } ab \wedge y \text{ right of } ab) \Rightarrow$$
$$\exists z : \text{Point}.\ x\_z\_y \wedge \text{Colinear}(z, a, b)$$

- Straightedge-Compass

$$(a \# b \ \wedge \ c\_b\_d) \Rightarrow$$
$$\exists u, v : \text{Point}.\ cu \cong cd \ \wedge \ cv \cong cd$$
$$\wedge \ a\_b\_u \ \wedge \ v\_b\_u \ \wedge \ \text{Colinear}(a, b, v) \ \wedge \ (b \# d \Rightarrow u \# v)$$

- Compass-Compass

$$(a \# c \ \wedge \ \text{Overlap}(a, b, c, d)) \Rightarrow$$
$$\exists u, v : \text{Point}.\ au \cong ab \ \wedge \ cu \cong cd \ \wedge \ av \cong ab \ \wedge \ cv \cong cd$$
$$\wedge \ \text{StrictOverlap}(a, b, c, d) \Rightarrow (u \text{ left of } ac \ \wedge \ v \text{ right of } ac)$$

The Constructive Axioms

- Co-transitivity of Apartness ("Magnifying Glass")

$$\forall a, b, c\!:\!\text{Point}.\ (a \# b) \Rightarrow (c \# a \vee c \# b)$$

- Non-triviality

$$\exists a, b\!:\!\text{Point}.\ a \# b$$

- The computational content of all theorems we prove comes from these two axioms, the three constructors, and the constructive interpretation of logic.
- Full axioms: 27
  Most don't have constructive content but allow us to reason about congruence and betweeness; we don't list them all here.

Computational Content

The computational content of any theorem will be expressed in the program Nuprl extracts from the proof; the content corresponds to the constructors, the constructive axioms, and the constructive interpretation of logic.

- **Co-transitivity of Apartness ("Magnifying Glass")**
  We define $M(a, b, c)$ as the function that decides whether $c \# a$ or $c \# b$.
- **Non-triviality**
  We refer to O and X as the two separated points in our Euclidean plane.

Computational Content

- **Straightedge-Straightedge**
  We let $SS(a, b, x, y)$ be the point z resulting from the Straightedge-Straightedge construction.
- **Straightedge-Compass**
  The SC axiom constructs from line ab and circle $C(c, d)$ points u and v, where u is on the opposite side of a from b and v is on the same side of b as a. So we define functions $\mathrm{SCO}(a, b, c, d)$ and $\mathrm{SCS}(a, b, c, d)$ to construct two points, SC+Opposite and SC+Same.
- **Compass-Compass**
  The CC axiom constructs a point $\mathrm{CCL}(a, b, c, d)$ to the left of ac and a point $\mathrm{CCR}(a, b, c, d)$ to the right of ac.

Proposition 1

To construct an equilateral triangle on a given finite straight line.

Euclid's first proposition is to construct an equilateral triangle on a given segment AB. We assume that $A\#B$ but include an extra property that the constructed point is to the left of AB (referring to the leftness relation on points). So the proposition becomes:

$$\forall A\!:\!\text{Point}.\ \forall B\!:\!\{\text{Point}|B\#A\}.$$
$$\exists C\!:\!\{\text{Point}|\ AB \cong BC\ \wedge\ BC \cong CA\ \wedge\ CA \cong AB \wedge C \text{ left of } AB\}.$$

We easily prove this as Euclid does by using the compass-compass (CC) axiom with circles $C(A, B)$ and $C(B, A)$. This constructs *two* equilateral triangles; we can choose to take only one such that C left of AB. Nuprl's extract of the constructive content of the proof is

$$\lambda A.\lambda B.\ \text{CCL}(A, B, B, A)$$

We then define $\triangle(A, B) = CCL(A, B, B, A)$ as the program for Euclid's proposition 1.

Proposition 2 (1 of 3)

> *To place a straight line equal to a given straight line with one end at a given point.*

Euclid's proposition 2 can be done with a ruler or a non-collapsing compass, so proposition 2 proves that a collapsing compass and a straightedge can construct these tools. It has been claimed that Proposition 2 is not constructively valid, but we can prove it from our axioms, using a slightly more complex proof than Euclid's.

Proposition 2 (2/3)

We use the construction depicted in the *Elements* as a lemma, requiring $A \# B$:

$$\forall A\!:\!\text{Point}.\ \forall B\!:\!\{\text{Point}|B \# A\}.\ \forall C\!:\!\text{Point}.\ \exists D\!:\!\{\text{Point}|AD \cong BC\}.$$

The program extracted from this lemma is

$$\begin{array}{c} \text{lmma2(A,B,C)} = \text{let X}{=}\triangle(A, B) \text{ in} \\ \text{let U=SCO(X,B,B,C) in} \\ \text{SCS(A,X,X,U).} \end{array}$$

Proposition 2 (3/3)

Proposition 2 in its full form is

$$\forall A, B, C : \text{Point}.\ \exists D : \{\text{Point} | AD \cong BC\}.$$

The program extracted from this lemma is

```
Prop2(A,B,C) = if M(O,X,A)
then  if M(A,O,B)  then  lmma2(A,B,C)
          else lmma2(AO,lmma2(O,B,C))
else  if M(A,X,B)  then  lmma2(A,B,C)
          else lmma2(A,X,lmma2(X,B,C)).
```