CDEG: Computerized Diagrammatic Euclidean Geometry 2.0

Nathaniel Miller

University of Northern Colorado, nat@alumni.princeton.edu

Abstract. This paper briefly describes **CDEG** 2.0, a computerized formal system for giving diagrammatic proofs in Euclidean geometry.

Keywords: diagrams, computer formal systems, Euclidean geometry

This paper briefly describes the computer proof system **CDEG**, version 2.0. **CDEG** stands for "Computerized Diagrammatic Euclidean Geometry." This computer proof system implements a diagrammatic formal system for giving diagram-based proofs of theorems of Euclidean geometry that are similar to the informal proofs found in Euclid's *Elements* [2]. It is based on the diagrammatic formal system **FG**, which is described in detail in my book, *Euclid and his Twentieth Century Rivals: Diagrams in the Logic of Euclidean Geometry* [11]. That book also describes an earlier version of **CDEG**; however, **CDEG** has evolved significantly since the publication of the book. In particular, a beta version of **CDEG** is now publicly available, and can be downloaded from `http://www.unco.edu/NHS/mathsci/facstaff/Miller/personal/CDEG/`. I encourage interested readers of this paper to download **CDEG** and to try it out for themselves.

When we say that **CDEG** is a diagrammatic computer proof system, this means that it allows its user to give geometric proofs using diagrams. It is based on a precisely defined syntax and semantics of Euclidean diagrams. To say that it has a precisely defined syntax means that all the rules of what constitutes a diagram and how we can move from one diagram to another have been completely specified. The fact that these rules are completely specified is perhaps obvious if you are using the formal system on a computer, since computers can only operate with such precisely defined rules. However, it was commonly thought for many years that it was not possible to give Euclidean diagrams a precise syntax, and that the rules governing the use of such diagrams were inherently informal.

To say that the system has a precisely defined semantics means that the meaning of each diagram has also been precisely specified. In general, one diagram drawn by **CDEG** can actually represent many different possible collections of lines and circles in the plane. What these collections all share, and share with the diagram that represents them, is that they all have the same topology. This means that any one can be stretched into any other, staying in the plane. So, for example, a diagram containing a single line segment represents all possible single line segments in the plane, since any such line segment can be stretched into any other. See [11] for more details concerning the syntax and semantics of Euclidean diagrams, and see [10] for more details of how to **CDEG** can be used, including a tutorial.

CDEG is essentially a computer implementation of the formal system **FG** described in [11]. Actually implementing the formal system on a computer was a highly non-trivial matter that took several years worth of work. Why would we want to implement an existing formal system on a computer?

The first reason that we might want a computer implementation is to demonstrate that this system really is completely formal: that the diagrams that are being manipulated are, indeed, completely specified as formal objects, and that the rules of the system are completely specified on these objects. With traditional, sentential formal systems, we do this by writing our axioms in a formal language, and then carefully writing rules of inference as typographical manipulations of sentences in this formal language. However, when our formal objects are diagrams, it is difficult to achieve this level of specificity without a computer implementation. Diagrams are complicated formal objects, and we have very strong informal intuitions about how they should work that may cloud our ability to judge if our rules have been completely formally specified.

Furthermore, even if our rules are completely formally specified, without a computer implementation, it will be quite difficult to play with the formal system to see what derivations are like, and to make sure that they really work the way that we think they will. This is particularly true in geometry, where constructions can lead to case branching, with a large number of cases that are virtually impossible to keep track of without using a computer. Thus, we may not be able to prove everything we think we can.

This worry is not just academic. Several other diagrammatic formal systems have been proposed by other researchers and have appeared in print but have later turned out to have ill-defined and/or unsound rules. For example, Isabel Luengo's formal system **DS1**, described in [5] and [4], turned out to be unsound, as explained in [11, Appendix C]. Likewise, John Mumma's **Eu**, described in [7], [8], [9], and, at a previous conference in the Diagrams series, in [6], is also unsound, as described in [12]. Neither of these proposed formal systems for geometry was implemented as a computer system, and neither worked quite in the way that their designers intended. Furthermore, both systems were examined by quite a number of article referees and dissertation committee members who failed to notice their significant problems. Thus, we should approach any proposed diagrammatic formal system with a certain amount of healthy skepticism. A working computer system is one way to allay some of this skepticism.

Secondly, a computer system is the only way to make a formal system widely available. Many potential users will not be able to make sense out of a formal system that is just specified mathematically, but will be able to try out a computer implementation.

The third reason for a computer implementation is to be able explore exactly what the formal system is able to prove. **CDEG**'s rules of inference are closely modeled on those found in Euclid's *Elements*[2], and I therefore claim that **CDEG** should be able to duplicate the first four books of Euclid's *Elements*. The only way to verify this claim is to systematically go through each of Euclid's proofs, and to see how to duplicate it within **CDEG**. To date, I have done this with many different proofs from Euclid's Book I, but have not yet gone systematically through all of Euclid's proofs. This is a future project of mine, and one that would be essentially impossible without the computer implementation.

As mentioned above, a previous version of **CDEG**, version 1.0, was discussed in [11], but was never made publicly available, because it did not include a stand-alone means of drawing its diagrams. The new version of **CDEG** relies on OGDF (the "Open Graph Drawing Framework") to lay out its diagrams, using the mixed model algorithm of Gutwenger and Mutzel [3].

Other significant changes to **CDEG** include the addition of the triangle congruence rules and rules for deleting pieces of diagrams, as well as numerous bug fixes. It also now has the ability to draw its own output diagrams rather than relying on an external program to do this. However, the version of **CDEG** that is now available is a beta version and most likely still contains bugs. If you try out **CDEG** and discover any bugs, please let me know by sending an email to `nat@alumni.princeton.edu`.

References

1. Euclid. 1956. *The Elements*. New York: Dover, 2nd edn. Translated with introduction and commentary by Thomas L. Heath.
2. Gutwenger, Carsten and Petra Mutzel. 1998. "Planar Polyline Drawings with Good Angular Resolution." *6th International Symposium on Graph Drawing 1998, Montreal (GD '98)*. Springer Lecture Notes in Computer Science 1547: 167-182.
3. Luengo, Isabel. 1995. *Diagrams In Geometry*. Ph.D. thesis, Indiana University.
4. Luengo, Isabel. 1996. "A Diagrammatic Subsystem of Hilbert's Geometry." In G. Allwein and J. Barwise, eds., *Logical Reasoning with Diagrams*. New York: Oxford University Press.
5. Mumma, John. 2008. "Ensuring generality in Euclid's diagrammatic arguments." In G. Stapleton, J. Howse, and J. Lee, eds., *Diagrammatic Representation and Inference: 5th International Conference, Diagrams 2008*. Springer Lecture Notes in Computer Science 5223: 222–235.
6. Mumma, John. 2006. *Intuition formalized: Ancient and modern methods of proof in elementary geometry.* Ph.D. Dissertation, Carnegie Mellon University. Retrieved May 20, 2010 from `http://www.contrib.andrew.cmu.edu/~jmumma/list.html`.
7. Mumma, John. 2010. "Proofs, pictures, and Euclid." *Synthese* Volume 175 (2): 255–287. DOI 10.1007/s11229-009-9509-9.
8. Mumma, John. 2008. "Review of Euclid and his twentieth century rivals: Diagrams in the logic of Euclidean geometry." *Philosophia Mathematica* 16(2): 256–264.
9. Miller, Nathaniel. **CDEG** *User's Manual.* Available from `http://www.unco.edu/NHS/mathsci/facstaff/Miller/personal/CDEG/`.
10. Miller, Nathaniel. 2007. *Euclid and his twentieth century rivals: Diagrams in the logic of Euclidean geometry.* Stanford, CA: CSLI Press.
11. Miller, Nathaniel. 2012. "On the Inconsistency of Mumma's Eu," *Notre Dame Journal of Formal Logic*, in press. Preprint available at `http://www.unco.edu/NHS/mathsci/facstaff/Miller/personal/diagrams`.