Real-time animated dynamic geometry in the classrooms by using fast Gröbner basis computations

Z. Kovács[1]

[1] *The Private University College of Education of the Diocese of Linz, Austria, zoltan@geogebra.org*

The third millennium renaissance of elementary geometry by harnessing software applications in the classroom is already a history. By introducing dynamic geometry systems like *The Geometer's Sketchpad*, *Cabri* or *Cinderella* the high number of mechanical numerical computations enabled students to make their own experiments with geometric objects easy.

Today's computers go one step forward. Not only fast numerical computations are available, but symbolical commands help, for instance, to expand or factorize expressions or solve equations or even equation systems.

Nevertheless, application of symbolic computations in dynamic geometry remained an unexplored field for many years. Recent researches in the direction to compute Gröbner bases significantly faster than earlier[1], opened the road to manipulate on tens of equation systems in several variables within a second. Thus now it is possible to create real-time animations based on purely symbolic computations. Such real-time animations include computing locus or envelope equations for geometry constructions on various software platforms like native Java programs, web applications or mobile/tablet solutions.

In this scientific work the state of the art will be demonstrated in the *GeoGebra* dynamic mathematics tool which fully connects computer algebra and dynamic geometry. By computing and visualizing the algebraic equation of a non-trivial locus 29 frame per second (FPS) can be reached on an Intel(R) Core(TM) i7-2620M CPU @ 2.70GHz desktop computer (8 GB RAM installed, Ubuntu Linux 14.04.4). The same benchmark reaches 13 FPS as a web application in Google Chrome 49.0.2623.110.

Classroom application of the used technology is mainly on discovering general properties of geometry diagrams. GeoGebra—like many other DGS tools—makes it possible to construct a figure by using objects like points, lines or circles, and then, for example, GeoGebra's **LocusEquation** command calculates the equation of a locus (given by the tracer and mover points) and plots that as an implicit curve. When the user drags the free points of the diagram, the locus equation will be automatically recomputed using heavy symbolic computations, but due to the effectivity of the calculations, the equation will be obtained very quickly. Finally the

[1] See [10, 9] for two studies among the open source computer algebra systems.

implicit curve will be plotted by using numerical methods, based on the symbolical form of the equation. (See [8] for a more detailed explanation.)

A similar scenario is to create the envelope equation of a set of output paths while the moving point is bound to another object. (An envelope is a curve that is tangent to each member of the family of the output paths at some point.) Obtaining the envelope is possible by using the **Envelope** command in GeoGebra. More on this topic can be found in [7].

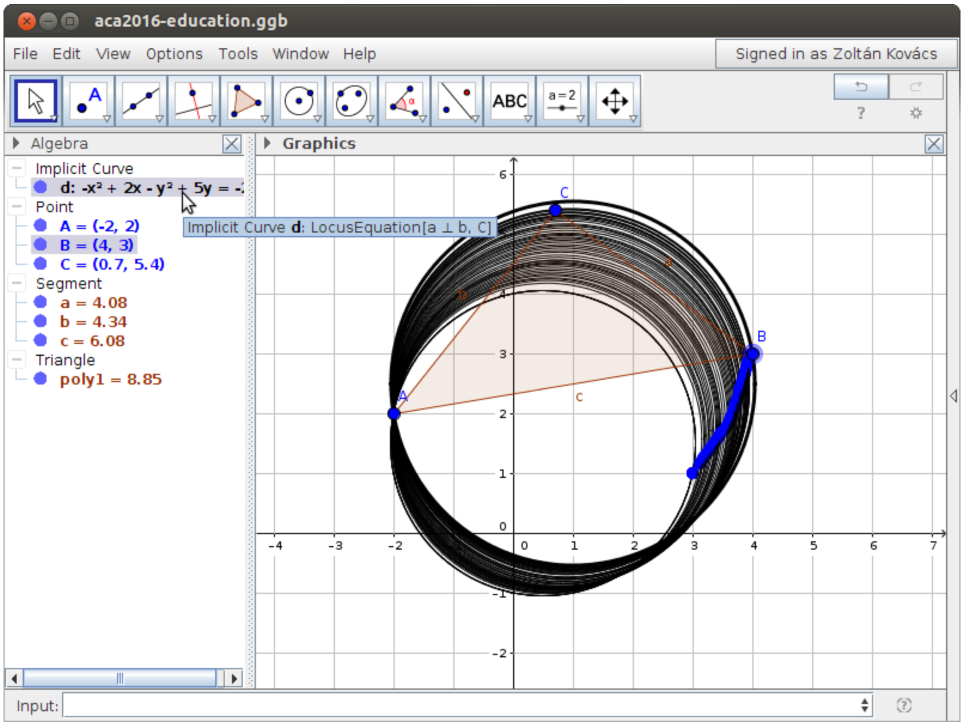
Recently another form of the **LocusEquation** command was introduced in GeoGebra, namely which calculates the locus of the free point such that a Boolean condition is satisified. This new form is also known as *implicit locus equation*. Due to the possibility to its direct use in the education, this third forms seems to be the most frequently used method to harness symbolic computations under the hood in a dynamic geometry experiment. For example, a natural way to discover Thales' theorem in the classroom is to construct a triangle *ABC* with GeoGebra, and then type the command `LocusEquation[`$a \perp b$`,`C`]`. In this way the circle with diameter *AB* will be immediately obtained and quickly redrawn when the free points *A* or *B* are dragged (Fig. 1).

All the three visualization activities above are common in using computational algebraic geometry to manipulate on coordinates of points, and their output is always an algebraic curve which is computed by—roughly speaking—eliminating some variables in the algebraic translation of the geometric construction. The elimination process is performed by the best available open source implementations, notably the standalone CAS *Singular* and the embedded system *Giac*.

The need to use symbolic computations instead of numerical ones in certain dynamic geometry visualization tasks is well discussed in the literature, see [5], section 3 for more details. All of the mentioned visualization activities are of the same kind.

The **LocusEquation** command was introduced in 2010 in GeoGebra's development version by the pioneer work of Botana's, Abánades', Escribano's and Arbeo's [4, 6], to translate the geometric construction into an algebraic equation system. The elimination step was implemented by outsourcing the computations to the computer algebra system *JAS* first, and later to *Reduce*, Singular and recently to Giac [10]. The visualization step of the locus equation as an implicit curve was implemented by Birklbauer and Drakulić [3] in 2011. Further improvements on the **LocusEquation** command were based on the joint work of the pioneers, and Montes' and Recio's [2].

Envelope support was added later by Botana and the author [5] in 2013. Implementation of the implicit locus equation was achieved by Abánades, Botana, Kovács, Recio and Sólyom-Gecse in 2016 (see [1] for a collection of examples).



Implicit Curve

- d: $-x^2 + 2x - y^2 + 5y = -$

Implicit Curve **d**: LocusEquation[a ⊥ b, C]

Point

- A = (-2, 2)
- B = (4, 3)
- C = (0.7, 5.4)

Segment

- a = 4.08
- b = 4.34
- c = 6.08

Triangle

- poly1 = 8.85

Figure 1: An imaginary classroom discovery of Thales' theorem while dragging point B from $(3,1)$ to $(4,3)$. Tracing is switched on for point B and implicit curve d. Here the animation reaches an average of 50 FPS for this simple theorem in desktop version of GeoGebra 5.0.237.0.

References

[1] M. Á. Abánades, F. Botana, Z. Kovács, T. Recio, C. Sólyom-Gecse, *Implicit loci*, http://tube.geogebra.org/m/mbXQuvUV (2016).

[2] M. Á. Abánades, F. Botana, A. Montes, T. Recio, *An algebraic taxonomy for locus computation in dynamic geometry*, Computer-Aided Design **56**, pp. 22-33 (2014).

[3] P. Birklbauer and D. Drakulić, *Implicit curves*. Second International GeoGebra Conference, Hagenberg, Austria. http://ggbconference2011.pbworks.com/w/file/fetch/44873718/presentation.pdf (2011).

[4] F. Botana and Z. Kovács, *Teaching loci and envelopes in GeoGebra*, http://tube.geogebra.org/m/R8nUbEQV (2014).

[5] F. Botana and Z. Kovács, *A Singular web service for geometric computations*, Annals of Mathematics and Artificial Intelligence, **74**, 3, pp. 359-370 (2015).

[6] F. Botana and Z. Kovács, *New tools in GeoGebra offering novel opportunities to teach loci and envelopes*. https://arxiv.org/abs/1605.09153, arXiv:1605.09153 [cs.CG] (2016).

[7] F. Botana and T. Recio, *Computing envelopes in dynamic geometry environments*, Annals of Mathematics and Artificial Intelligence, First online: 01 March 2016, pp. 1-18 (2016).

[8] Z. Kovács, *Loci problems in Age of Reason and their effect on GeoGebra – Locus equations and their factorization in GeoGebra*. International GeoGebra Conference, Budapest, Hungary. http://test.geogebra.org/~kovzol/talks/igi14/bp.pdf (2014).

[9] Z. Kovács, *Computer based conjectures and proofs in teaching Euclidean geometry*. PhD thesis, Johannes Kepler University, Linz, Austria (2015).

[10] Z. Kovács and B. Parisse, *Giac and GeoGebra – Improved Gröbner basis computations*, in *Computer Algebra and Polynomials, Volume 8942 of the series Lecture Notes in Computer Science*, pp. 126-138 (2015).