Geometry Expressions: A Constraint Based Interactive Symbolic Geometry System

Philip Todd

Saltire Software, POB 1565 Beaverton OR,
U.S.A.
philt@saltire.com

Abstract. Real Euclidean geometry is a basic mathematical dialect, not only of high school students, but also of mechanical engineers, graphics programmers, architects, surveyors, machinists, and many more. In this paper, we present "Geometry Expressions": an interactive symbolic geometry package. The aim of the software is to generate algebraic formulas from geometry. It is a further intention of the software that the model should be entered interactively in a style which is convenient to both the geometry consumer groups identified above.

1 Introduction

Interactive Geometry Systems and Computer Algebra Systems both have an established place in education, the former more heavily at the high school level, the latter at the college level. The importance of an interactive symbolic geometry package is that it constitutes a bridge between these two areas of technology: geometry can be entered graphically, symbolic expressions output which may then be transferred to an algebra system for further analysis. It is natural for both consumer groups to merge geometric descriptions with algebraic: the tree that casts a shadow has a height h, and the shadow a length s, a family of mechanical parts is parameterized by exterior and interior diameters D, and d. The author is not aware of any existing software which enables the convenient coexistence of the symbolic with the geometric.

Previous work linking interactive geometry and algebra systems include The Algebraic Geometer, Geometry Expert, paramGeo and Geother and GDI [1-4]. The flavor of these systems is to use a link to an algebra system to prove geometry theorems posed in a dynamical geometry context. Our focus, in contrast, is not on theorem proving per se, but on formula generation. A traditional dynamic geometry system is not the best format for the user interface of a formula generation package, as it does not provide particularly convenient ways to attach symbolic inputs to a model. Quantities are typically derived from locations instead of the other way round. (Distances can, nevertheless, be specified as parameters of translations, angles as parameters of rotations). A constraint based model, however, allows such quantities as distances and angles to be specified directly. This is a much more natural style of user interface for a formula generation package.

F. Botana and T. Recio (Eds.): ADG 2006, LNAI 4869, pp. 189–202, 2007.

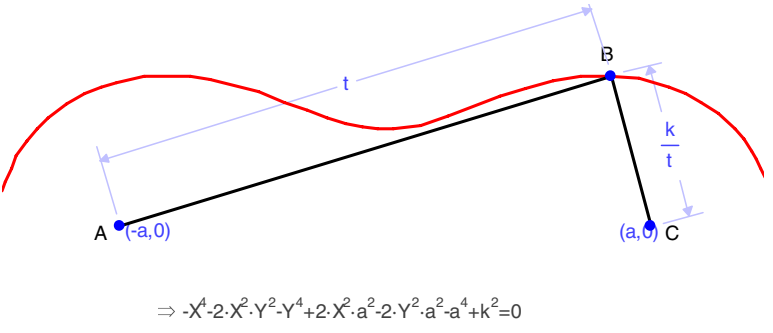

Fig. 1. Oval of Cassini defined in terms of a pair of distance constraints

For example, Cassini Ovals are defined as the loci of points the product of whose distance from two fixed points is constant. This is naturally expressed as a pair of symbolic distance constraints from points whose location on a coordinate plane have also been symbolically constrained (Fig. 1). Given these inputs, Geometry Expressions can output the implicit equation of the resulting locus as an expression whose coefficients depend on the undetermined symbols k, the product of the lengths, and a, the absolute value of the x coordinates of the foci.

In this paper, we describe the overall architecture of Geometry Expressions, detail some aspects of the system design, and illustrate through examples the usage of the software.

2 System Architecture

The geometry engine in Geometry Expressions works in the following way [5]

1. A sketch of the geometry along with a set of symbolic constraints is entered by the user.
2. Graph algorithms [6] are used to convert the constraint based description into a sequence of elementary constructions.
3. The construction sequence is executed symbolically resulting in algebraic expressions for the location of each of the geometric objects in the drawing.
4. Measurements made from the drawing are converted into algebraic expressions involving the locations of the geometry objects. The expressions thus obtained are simplified using standard techniques of computer algebra, along with some geometry specific heuristics, and presented to the user.

For example, the drawing of a triangle constrained by two sides and the included angle in (Fig. 2) is converted into this construction sequence:

1. Create a point A at arbitrary location.
2. Create a line AB through A with arbitrary direction.
3. Create a point B on line AB distance a from point A.

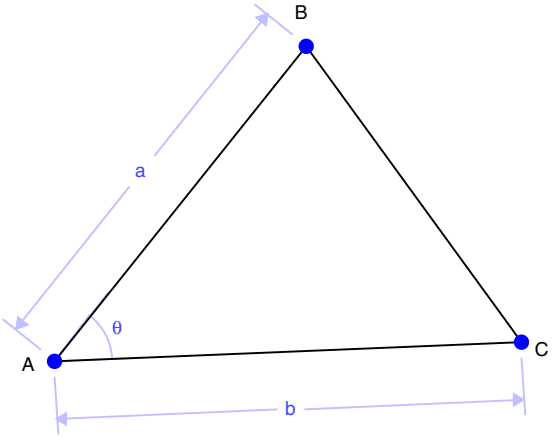

Fig. 2. A geometric figure is specified by a sketch with symbolic constraints (algorithm step 1)

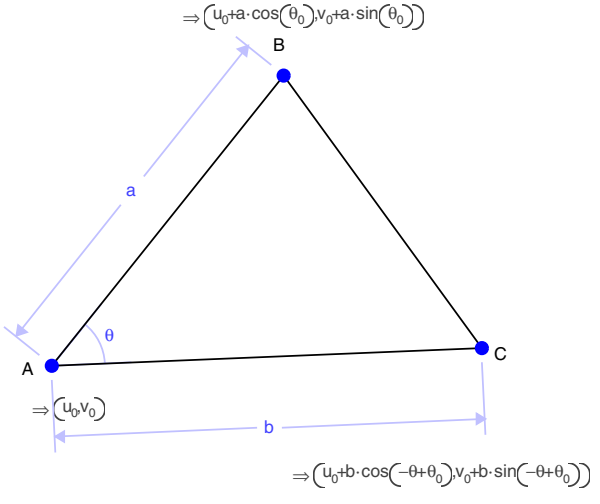

Fig. 3. Geometry Expressions computes symbolic locations for points in a constrained triangle (algorithm step 3)

4. Create a line AC through A and with direction angle θ from line AB.
5. Create a point C on line AC distance b from point A.

Given this construction sequence, Geometry Expressions creates locations for all the points (Fig. 3). Where there is any freedom in the model, Geometry Expressions adds system-generated variables. For example, in fig. 3, the location of A is arbitrary, and the system adds variables u_0 and v_0 as its coordinates. It also adds the variable θ_0 for the arbitrary direction AB.

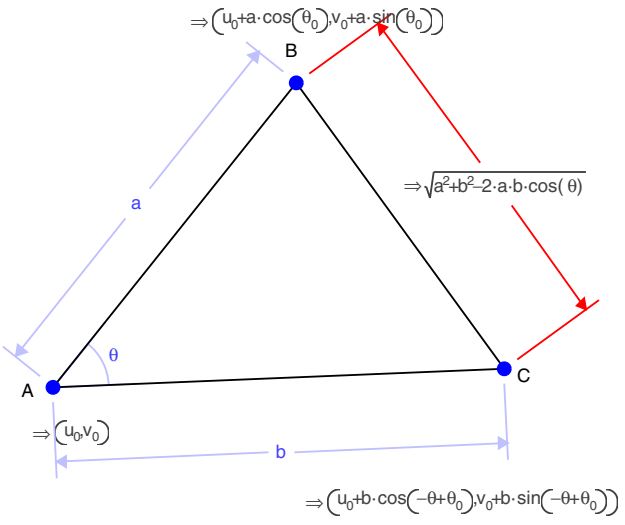

Fig. 4. Output measurements are computed and simplified (algorithm step 4)

When the user asks for a measurement from the drawing, the equivalent algebraic expression is evaluated, simplified and presented. For example, if he asks for the distance between B and C, the distance formula is applied to the symbolic expressions for the coordinates of A and B, simplified and displayed on the diagram (fig. 4).

Within this overall architectural framework, there are a number of design features which are essential to the practicality of the system. We will discuss the following features:

- Intermediate variable retention
- Use of real witness values for variables
- Use of MathML to facilitate two way communication with algebra systems.

2.1 Intermediate Variable Retention

In a purely numerical system, the space and time requirements of the above algorithm are linear in the number of primitive geometric entities. In a symbolic implementation, however, the size of algebraic expressions grows exponentially in the length of the construction sequence. The linear characteristics of the numerical algorithm can be recaptured in the symbolic domain by creating geometric intermediate variables and retaining them in the symbolic representation of the model, and initially in the output measurements.

The user is able to control the display of intermediate variables, by specifying whether they should be retained, and by setting a global granularity parameter. The system substitutes away intermediate variables whose definition is deemed too simple (for example, an intermediate variable which is defined to

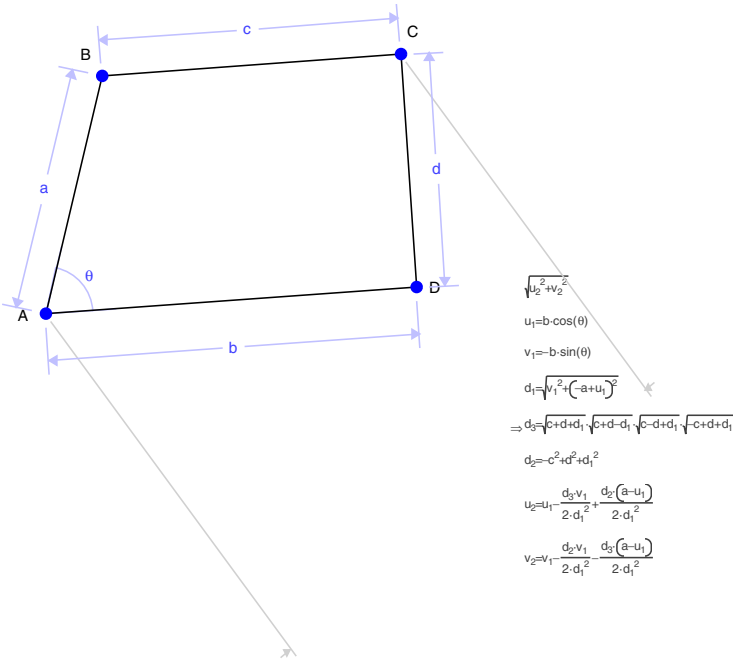

Fig. 5. Distance AD with fine grained intermediate variables. The intermediate variables u_1, v_1, u_2, v_2, d_1, d_2, d_3, are local to the expression and defined, ultimately, in terms of the input variables a,b,c,d, θ.

be a constant would always be considered too simple to retain). The granularity parameter controls the definition of "too simple". Figures 5 and 6 show the same measurement with different settings for the granularity parameter. In Fig. 5, the parameter is set "fine" with the result that more intermediate variables are retained, but their definitions are simple. In Fig. 6, the granularity parameter is set coarser with the result that fewer, but more complicated, intermediate variables are present.

2.2 Witness Values for Variables

The definition of a problem in Geometry Expressions has two components. Algebraically a problem comprises a set of constraints between entities. These constraints correspond to symbolic expressions. In addition, the problem definition contains a sketch of the intended geometry. In general there may be more than one solution to the set of equations corresponding to the constraints. The sketch of the intended geometry is used to choose which solution to use.

In Fig. 7, for example, both triangles ABC and ADC are defined by the same set of constraints (two sides and the non-included angle) however, because ABC is sketched as an acute angle, and ADC is sketched as an obtuse angle,

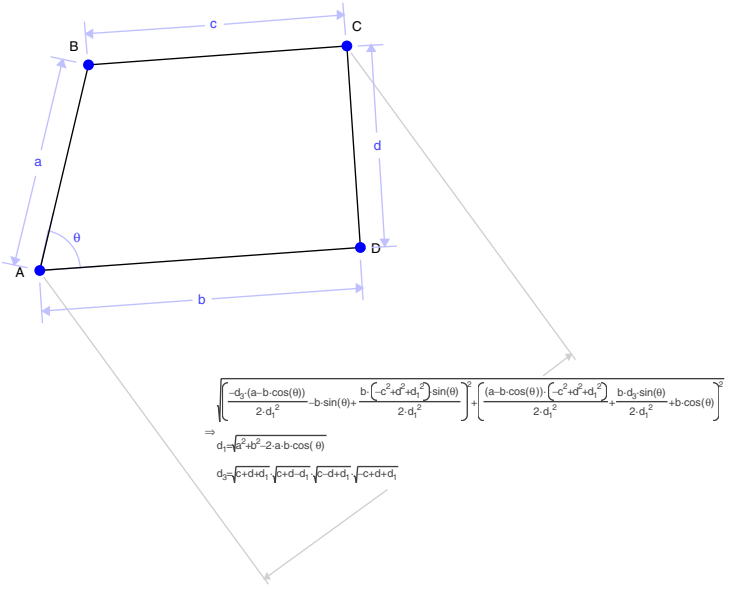

Fig. 6. The same measurement as Fig. 5, but with a coarser setting to the intermediate variable granularity parameter

Geometry Expressions has resolved B and D to different locations - and the symbolic outputs for the lengths BC and DC are indeed different. To reiterate, in terms of symbolic constraints (i.e. algebra) triangles ABC and ADC are defined identically. It is the drawing (i.e. geometry) which leads the symbolic values of BC and DC to differ.

Each solution to the symbolic constraint set represents a family of numerical solutions. Geometry Expressions displays a representative member of the family. In order to do this, it needs to substitute a real number for each of the input variables. For example, in Fig. 7 AB and AD are both specified to have length a. In order to draw a representative solution, the system needs a numeric value for a. In theory, a could have any of a wide range of values, however in practice, the user expects that the representative member of the solution family chosen by the system should be fairly close to his original sketch.

We call the specific numeric value used in the sketch the witness value for the variable. Geometry Expressions has a subsystem for deriving witness values from the sketch, and maintaining witness values throughout its algebra system. Constraints may be specified as expressions involving input variables (Fig. 1, for example). To derive plausible witness values for these inputs, Geometry Expression uses a general purpose numeric root finder. In addition there is a user interface subsystem which allows the user to explicitly set witness values, and to control their behavior on dragging.

A further use of witness values for variables is to allow the automatic creation of assumptions in order to simplify output expressions involving absolute values.

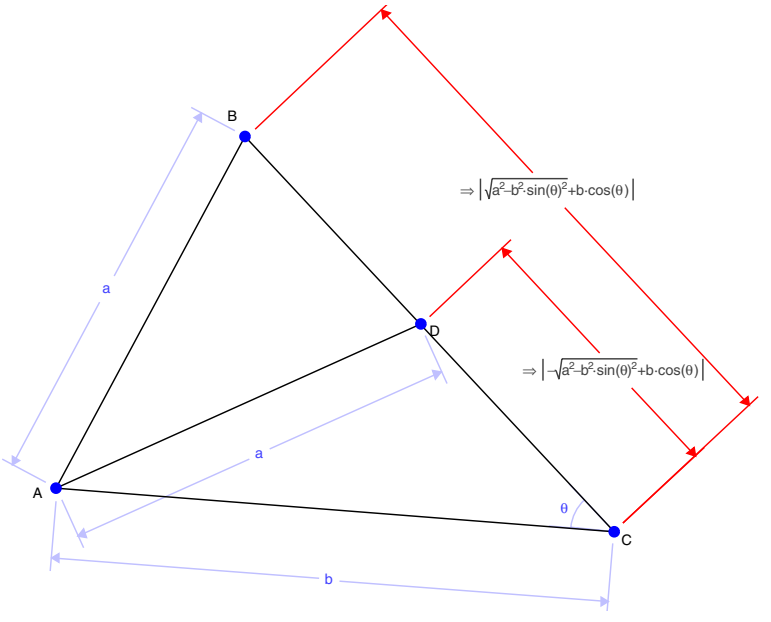

Fig. 7. Triangle ABC is specified in terms of two sides and the non included angle. There are of course two possible such triangles (ADC is the other one). The solution branch which contains the witness triangle is selected by the application.

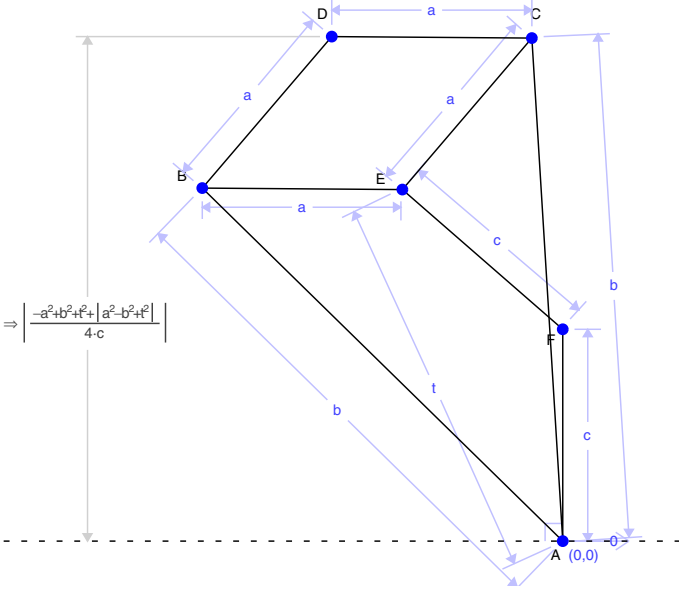

Fig. 8. Paucellier's linkage with the height of D displayed without invoking assumptions based on witness values

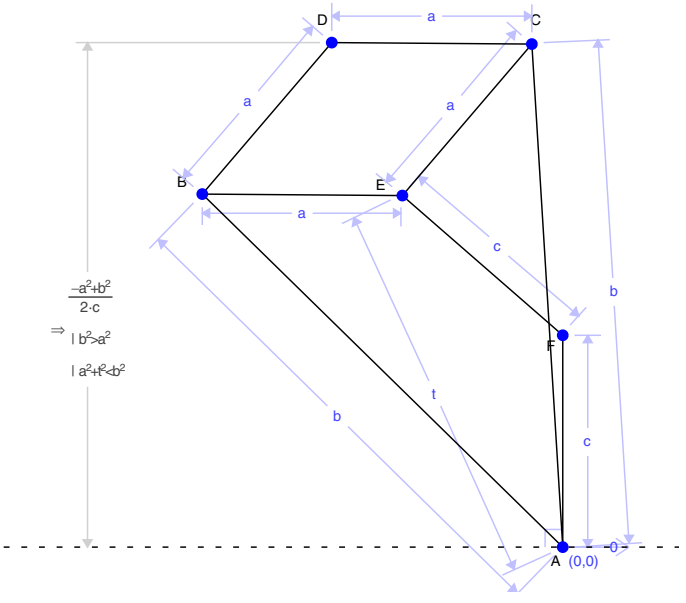

Fig. 9. The height of D simplified with explicit assumptions automatically supplied based on witness values for a,b, and t

Geometry Expressions has both an implicit and an explicit assumptions mechanism. Implicit assumptions are derived from the knowledge that any variables which are used to specify distances or radii must be positive. If the argument of an absolute value can be deduced to be strictly positive or strictly negative based on this information, then the absolute value can be simplified. Such implicit assumptions are applied automatically.

Explicit assumptions may be applied at the user's request. In this case, a real numeric value for the argument of an absolute value is determined based on witness values for any variables which are present. The absolute value is replaced by its argument or by the negative of its argument depending on the sign of this numeric value. An explicit assumption is added to the output expression.

Figures 8 and 9 both show models of Paucellier's linkage. The design of this linkage is such that varying the distance t between B and F should move D in a horizontal straight line. In Fig. 8, the fact that the height of D is independent of t is obscured by the absolute values. In Fig. 9, the addition of assumptions based on witness values for the variables makes its independence of t clear.

2.3 MathML

Geometry Expressions has a special purpose algebra system built in, which is responsible for maintaining and simplifying the expressions generated by the geometric models. However it does not contain a full general purpose CAS. Instead there is a capability for importing and exporting MathML. This allows

the user to copy symbolic measurements from Geometry Expressions into the algebra system of his choice, perform some analysis, then, if appropriate, paste the results back into Geometry Expressions.

MathML Example. As an illustration of the use of Geometry Expressions in conjunction with an algebra system, we describe an investigation of the location of the cusps observed in the caustic formed by light from a finite point source reflecting in a cylinder.

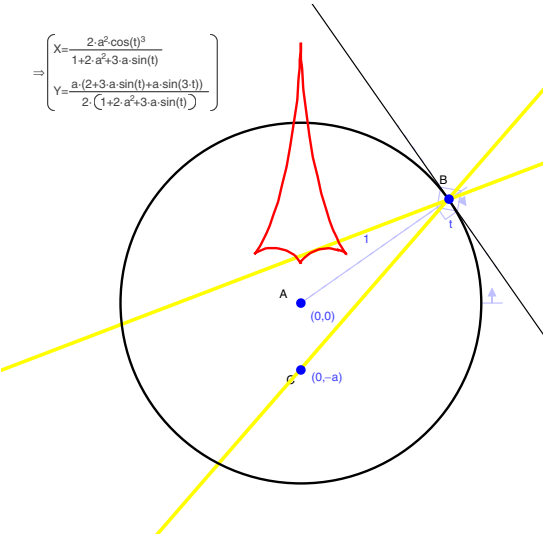

Fig. 10. Parametric equation of the envelope of the rays emanating from the point D (0,-a) and reflected in the unit circle centered at the origin. The parameter t corresponds to the direction AB.

To model this situation in Geometry Expressions (fig 10), we create a circle, AB and an infinite line through B. We constrain the center of the circle to have coordinates (0,0), and constrain its radius to be 1, and we constrain the line through B to be tangential to the circle. We constrain the parametric location of B on the circle to be t. This has the effect of setting the angle between AB and the x axis to be t. We then create a point D and constrain its coordinates to be (0,-a), create a line through D and B, and construct its reflection in the tangent line. Finally we create the envelope of the reflected line as t varies between 0 and 2π.

Two cusps of the envelope curve clearly lie on the y axis and are obtained when B is at parametric locations $\frac{\pi}{2}$ and $-\frac{\pi}{2}$. Creating points on the envelope curve at these parametric locations, we can observe that they do indeed lie at the cusps. Geometry Expressions computes their coordinates.

The parametric location of the other cusps is not so obvious. To compute these, we use the facility of copying and pasting to and from an algebra system

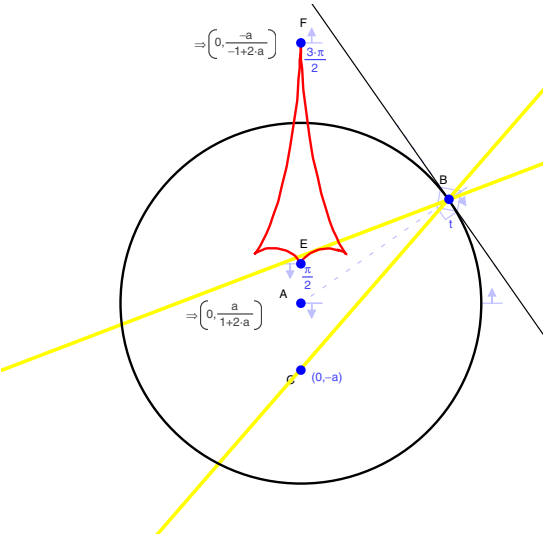

Fig. 11. Coordinates for the points at parametric locations $\frac{\pi}{2}$ and $-\frac{\pi}{2}$ on the envelope curve

(in this case Maple) via MathML. Copying the curve equation into Maple, we differentiate and then solve for the derivatives being simultaneously 0.

$$x := \frac{2\cos(t)^3 a^2}{1 + 2a^2 + 3\sin(t)a} \tag{1}$$

$$y := \frac{1}{2}\frac{(2 + 3\sin(t)a + \sin(3t)a)a}{1 + 2a^2 + 3\sin(t)a} \tag{2}$$

> solve({diff(x,t)=0,diff(y,t)=0},t);

$$\{t = -\frac{\pi}{2}\}, \{t = \frac{\pi}{2}\}, \{t = arctan(-a, RootOf(\_Z^2 - 1 + a^2))\} \tag{3}$$

> allvalues(t = arctan(-a,RootOf(_Z^2-1+a^2)));

$$t = arctan(-a, \sqrt{1 - a^2}), t = arctan(-a, -\sqrt{1 - a^2}) \tag{4}$$

Putting the arctan as the parameter value on a point on the curve, we can have Geometry Expressions give the coordinates of the point (fig 12)

We notice in this example, the use of the two way communication between Geometry Expressions and the algebra system. Curve equations generated from the geometrical problem were exported to the algebra system in order to be differentiated and solved to find cusp locations. Cusp locations were then copied back into Geometry Expressions for further geometrical analysis. The two way communication between these tools makes for a very productive learning and research environment.

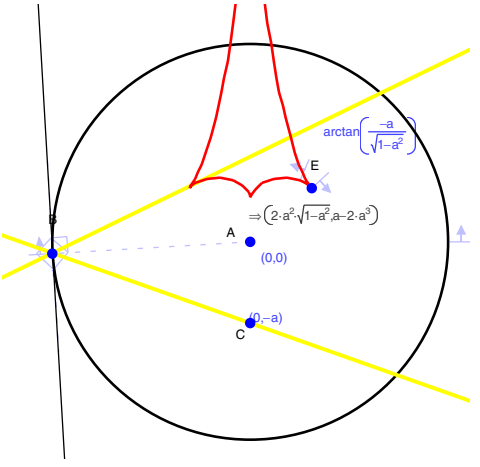

Fig. 12. Coordinates of the cusp, whose parametric location on the envelope curve corresponds to the solution found in Maple

3 Theorem Proving

The interactive nature of the use of Geometry Expressions makes it a useful tool for the discovery as well as the automatic proof of theorems. We illustrate this with an example involving mixtilinear incircles and excircles.

A mixtilinear incircle is tangent to 2 sides of a triangle and (internally) to the circumcircle. A mixtilinear excircle is tangent to 2 sides of a triangle and (externally) to the circumcircle. We show that the ratio of the radii of the mixtilinear excircles and the mixtilinear incircles satisfy an analogous relationship to that between the incircle and excircles.

In Fig. 13, we have constrained a triangle by specifying its side lengths, and created one mixtilinear incircle/excircle pair. Geometry Expressions computes the radii of these circles in terms of the side lengths of the triangle. Observing a degree of commonality between the radii, we display their ratio.

Observing that the ratio displayed has a numerator which is symmetric in a,b,c we are led to consider the sum of the reciprocals of the 3 such ratios. That is the sum of the mixtilinear incircle/excircle radii. Simple algebra leads to the result that if r1, r2, r3 are the radii of the mixtilinear incircles, and if s1, s2, s3 are the radii of the mixtilinear excircles, then:

$$\frac{r_0}{s_0} + \frac{r_1}{s_1} + \frac{r_2}{s_2} = 1 \tag{5}$$

This is analogous to the relationship between the incircle radius and the excircle radii [7]:

$$\frac{1}{s_0} + \frac{1}{s_1} + \frac{1}{s_2} = \frac{1}{r} \tag{6}$$

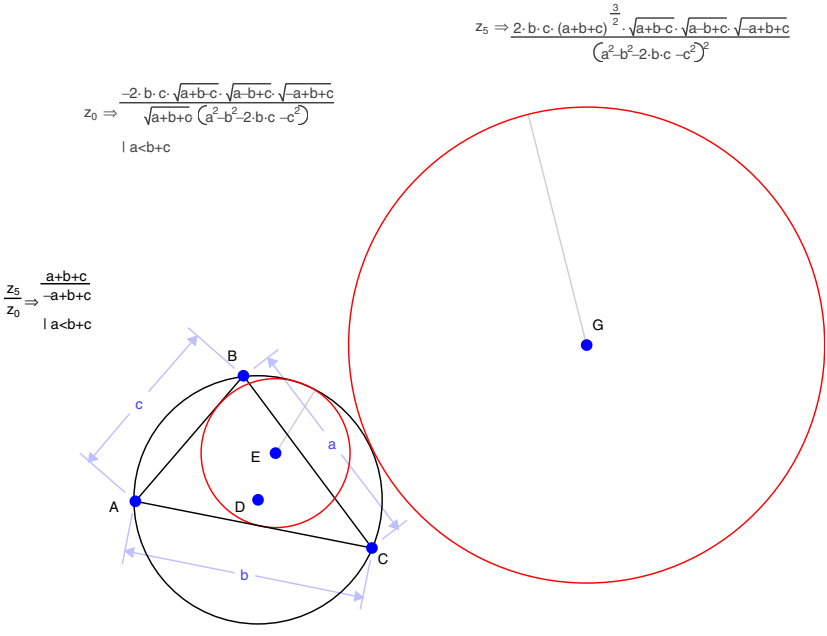

Fig. 13. Expression for the coordinates of the center of the incircle in terms of the coordinates of the triangle vertices

While we make no claims that this result is new in an absolute sense, it was certainly new to us, and its "discovery" facilitated by the formula generation capabilities of Geometry Expressions, working in collaboration with a little human pattern matching.

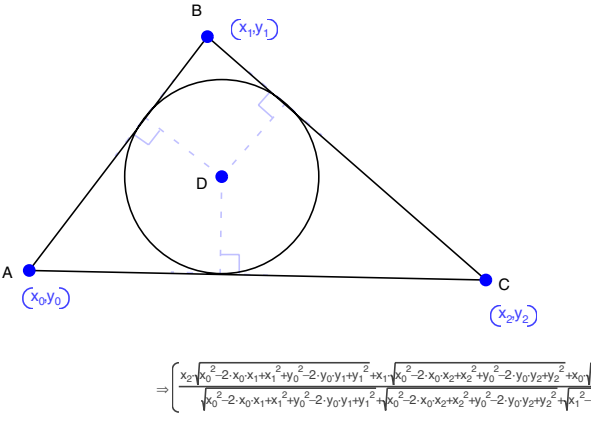

Fig. 14. Expression for the coordinates of the center of the incircle in terms of the coordinates of the triangle vertices

4 Further Work

One shortcoming of the approach lies in the fact that the user only has an opportunity to specify symbolic values for an independent set of constraints. In some situations, the form of the symbolic output could be improved by the addition of names for dependent quantities.

For example (Fig. 14), the coordinates of the incenter of a triangle expressed in terms of the vertex coordinates are cumbersome expressions. However, a cursory inspection shows that the terms under the roots are all the distance formula for the lengths of the sides. The complexity of the expression could then be significantly improved if the side lengths are named (Fig. 15).

A topic of further investigation is to extend the basic model so that the user may specify a dependent set of geometric variables.

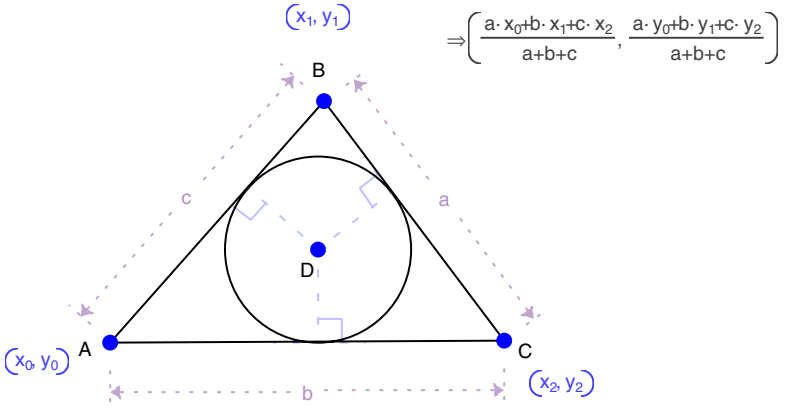

Fig. 15. Expression for the coordinates of the center of the incircle in terms of the coordinates of the triangle vertices and the side lengths

5 Conclusion

A constraint based interactive symbolic geometry system such as Geometry Expressions facilitates a collaborative approach to automated geometry. Collaboration between geometry system and algebra system is enabled by MathML based communication and illustrated by the light caustic example of fig. 10. In that example, the geometry system was used to generate an equation for the caustic curve. The algebra system was used to calculate the location of cusps, and those locations fed back into the geometry system, which was used to display their locus, and to derive the equation of the locus curve.

Collaboration between user and computer is illustrated in the mixtilinear incircle/excircle example of fig.13. The computer is used to generate equations for circle radii in terms of triangle side lengths. Examination and manipulation of these results by a human user leads to the "discovery" of a theorem relating the radii.

References

1. Chou, S.C., Gao, X.S., Ye, Z.: Java Geometry Expert. In: Proceedings of the 10th Asian Technology Conference in Mathematics, pp. 78–84 (2005)
2. Lozano, E.R., Macáas, E.R., Mena, M.V.: A Bridge Between Dynamic Geometry and Computer Algebra. Mathematical and Computer Modelling 37(9-10), 1005–1028 (2003)
3. Lozano, E.R.: Boosting the Geometrical Possibilities of Dynamic Geometry Systems and Computer Algebra Systems Through Cooperation. In: Borovcnik, M., Kautschitsch, H. (eds.) Technology in Mathematics Teaching. Proceedings of ICTMT– 5. öbv & hpt, Schrifrenreihe Didaktik der Mathematik, Viena, vol. 25, pp. 335–348 (2002)
4. Wang, D.: GEOTHER 1.1: Handling and Proving Geometric Theorems Automatically. In: Hong, H., Wang, D. (eds.) ADG 2004. LNCS (LNAI), vol. 3763, pp. 92–110. Springer, Heidelberg (2006)
5. Todd, P., Cherry, G.: Symbolic analysis of planar drawings. In: Gianni, P. (ed.) Symbolic and Algebraic Computation. LNCS, vol. 358, pp. 344–355. Springer, Heidelberg (1989)
6. Todd, P.: A k-tree generalisation that characterises consistency of dimensioned engineering drawings. SIAM Journal of Discrete Mathematics 2, 255–261 (1989)
7. Weisstein, E.W.: "Excircles." From MathWorld–A Wolfram Web Resource, `http://mathworld.wolfram.com/Excircles.html`