Robust geometry for a graphics language

Henrik Tidefelt
tidefelt@isy.liu.se

May 21, 2007

Abstract

As a hobby, I design and implement a computer language for the creation of 2D and 3D graphics. The graphics are "vector based", and in order to get seamless integration of 2D and 3D graphics, the 3D objects must sooner or later be projected on the 2D canvas. The challenge comes from the demand that the projection shall be represented in 2D. The current implementation is not sufficiently robust to handle any but the smallest examples, and this project proposal is to make a replacement.

Background

Some years ago, I started using the old graphics language *MetaPost* [Hobby, 1994] to make illustrations for a wide range of applications. I was excited about the power of using a computer language to draw pictures, but after a while I was both tired of the low level of the language and found several important abstractions missing. This is not to say that MetaPost was poorly designed, but just that it is a product of its time — since then, computers have become faster, and it has become feasible to work with higher level languages, and operations that used to be too costly to be part of the language are now affordable.

There are existing languages designed to replace MetaPost, perhaps most notably *Asymptote* [Hammerlindl et al., 2004]. Several graphics languages are built on top of LaTeX, including *PGF* and *TikZ* [Tantau, 2005], but these are not really comparable since they cannot afford to work with abstractions that require extensive numerical computation.

The work on the language for this project, under the working name *Drool*, was initiated some time during 2005 (the initial import to the Subversion repository was in September that year). By then, I was not aware of Asymptote, perhaps because it was not mature enough to be promoted at that time. Had I known about Asymptote, chances would have been small that the work on Drool had begun, but looking back I am content with how things turned out, because I really consider Drool a potential alternative (not replacement!) to Asymptote. Asymptote is an imperative language. Drool is not.

Today, Drool has been used in the production of a printed book, I use it in an even wider range of applications than was possible with MetaPost, and there is a variety of examples available that show features of the language.

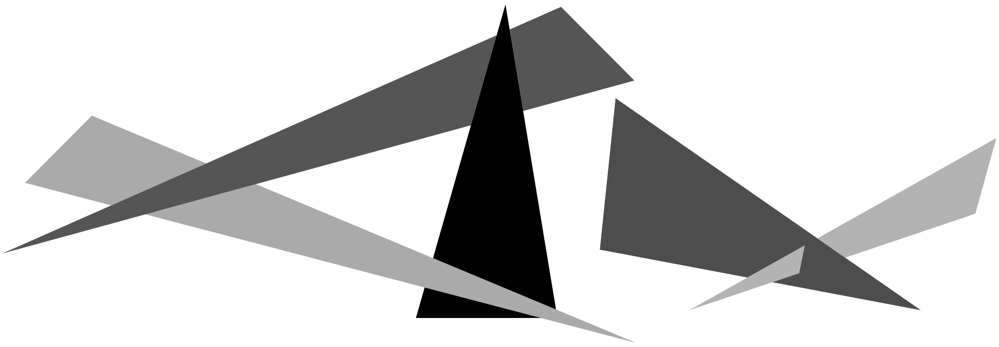

Figure 1. Correct view of overlapping and intersection objects in 3D.

Some important tasks for the future are

- Addition of an optional type system.
- Compilation of mathematical shading functions to the PostScript calculator used in PDF.
- Improvements of numerical algorithms.
- Testing and debugging.
- Language documentation and a user guide.

Background more specific to this project concerns the 3D graphics of Drool. Traditionally, as far as I know, 3D graphics are projected to 2D in two ways. The most common is to wait until the raster of the output device is known, and then render the 3D graphics in a z-buffer which incrementally keeps track of the distance to the currently closest object at each raster point. The vector-based alternative is to sort all objects according to their distance to the eye, and then draw their 2D projections in back-to-front order. Note that the latter technique really is vector-based since the 2D projections does not have to be rasterized. The fallacy of this approach, however, is that it is not always possible to order the objects due to cyclic overlaps and intersections, see figure 1. Although figure 1 is created using Drool, the implementation is not robust enough to scale to bigger problems. In addition, the current design was not made with shadows in mind; the problem of computing (sharp) shadows is closely related, and should be solved using the same algorithm.

Project proposal

The overall goal is to create a viable replacement for the current implementation of the projection of 3D objects to 2D. The algorithm shall be applicable also to the computation of sharp shadows. Emphasis shall be on robustness and scalability. This should be achieved using geometrical invariants in the design of the algorithm.

Students that take on this project should have a strong interest in algorithm correctness and geometry. The project outcome should be rewarding as the result is visual by nature.

The implementation language is C++, and a silly lines-of-code count of the current implementation adds to about 3500 (about 5% of the project total). A clever implementation may need less.

As Drool is a non-commercial project, no salary will be given. No special hardware or software is required; today, the tools used are GCC 3.4.6 or GCC 4.0.0, flex 2.5.4, bison 2.3, GNU Make 3.80, and Subversion 1.3.1. Dependencies include GSL, BLAS, iconv, and zLib.

References

Andy Hammerlindl, John Bowman, and Tom Prince. *Asymptote: the vector graphics language*, version 1.29svn edition, 2004. URL `http://asymptote.sourceforge.net/`.

John D. Hobby. *A user's manual for MetaPost.* AT&T Bell Laboratories, Murray Hill, NJ 07974, 1994. URL `http://cm.bell-labs.com/who/hobby/MetaPost.html`.

Till Tantau. *TikZ and PGF — Manual for version 1.00*, 2005. URL `http://sourceforge.net/projects/pgf/`.