An Extension of CGAL to the Oriented Projective Plane $\mathbb{T}^2$ and its Dynamic Visualization System

A. G. Oliveira *
Institute of Computing
Universidade Estadual de Campinas
Campinas, Brazil

P. J. de Rezende
Institute of Computing
Universidade Estadual de Campinas
Campinas, Brazil

F. P. Selmi-Dei †
Institute of Computing
Universidade Estadual de Campinas
Campinas, Brazil

rezende@ic.unicamp.br

ABSTRACT

The oriented projective plane $\mathbb{T}^2$ is an extension of the Euclidean plane $\mathbb{E}^2$ and comprises a number of advantages for algorithm design and implementation. We have extended the Computational Geometry Algorithms Library (CGAL) to allow for the implementation of geometric primitives and algorithms. The present video illustrates both the extension of a few algorithms to $\mathbb{T}^2$ under CGAL and a dynamic visualization system ($\mathbb{T}^2$Viewer) built specially for displaying the spherical and planar models of $\mathbb{T}^2$.

Categories and Subject Descriptors

I.3.5 [**Computational Geometry and Object Modeling**]: [Geometric algorithms, languages, and systems]

General Terms

Algorithms

Keywords

Oriented Projective Plane, CGAL, Visualization, Algorithm Animation

1. BACKGROUND

The oriented projective plane $\mathbb{T}^2$, which is an extension of the Euclidean plane $\mathbb{E}^2$, can be viewed by means of two very insightful models. By representing each point with signed homogeneous coordinates $[w, x, y] \in \mathbb{R}^3_*$, one naturally arrives at the planar model of $\mathbb{T}^2$ which consists of two copies of $\mathbb{E}^2$, representing the two ranges of $\mathbb{T}^2$ ($w > 0$ and $w < 0$), and a circle S^1, representing the points at infinity ($w = 0$). An equally natural visualization of $\mathbb{T}^2$ is the spherical model which consists of the surface of the sphere S^2 whose points (w, x, y) belong to the front range when $w > 0$, to the back range when $w < 0$ and to the line at infinity when $w = 0$.

Among the notions that require being redefined is that of the distance between points on $\mathbb{T}^2$. Since there are pairs of points at non-finite distance, we need to consider a space larger than $\mathbb{R}$ as the range of the distance function. The oriented projective line is the obvious choice. Thus, the square of the distance between two points $p = [p_w, p_x, p_y]$ and $q = [q_w, q_x, q_y]$ is given by:

$$(d_{\mathbb{T}^2_*}(p,q))^2 = [\text{sign}(p_w q_w).(p_w q_w)^2, (p_x q_w - q_x p_w)^2 + (p_y q_w - q_y p_w)^2].$$

In the context of computational geometry, the oriented projective plane (OPP) has been extensively studied. The interested reader is referred to [8, 1, 2] . Among the advantages of the two dimensional oriented projective geometry are that homogeneous coordinates result in simpler formulas; representation of and operations with (improper) points at infinity are as natural as with proper points ($w \neq 0$); there is a perfect duality between points and lines; and orientation of triples of points and convexity are well defined [8, 1].

However, to our knowledge, only [4], [6] and [9] worked on implementation of geometric algorithms on $\mathbb{T}^2$ (on an IRIX/SGI proprietary platform). Our goal was to provide a workbench for development of geometric algorithms which would be broadly accessible. Our choice of working environment was CGAL, the widely used Computational Geometry Algorithms Library [3]. To achieve the goal of producing a fairly easy to use environment for implementation of algorithms on $\mathbb{T}^2$, we extended the necessary components of CGAL.

2. EXTENSION OF CGAL TO THE ORIENTED PROJECTIVE PLANE

CGAL has been developed using the Generic Programming paradigm in C++ which lead our effort to the same approach [3, 5]. In order to allow for the implementation of algorithms on $\mathbb{T}^2$, firstly, geometric primitives and predicates were extended to work with signed homogeneous coordinates. Later, we also extended (Euclidean) algorithms to work on the OPP which showed that a significant number of special cases, dealt with on the Euclidean plane, did not need separate treatment in the extended algorithms.

The extension of the following primitives and predicates have been included in CGAL: points (proper and improper), straight lines, segments, rays, vectors, triangles, isothetic rectangles, bounding boxes and circles; distance computation (finite and infinite, in the sense of $\mathbb{T}^1$) and comparisons (absolute and relative, among proper and improper points), orientation of triples of points, point-segment incidence, relative position with respect to lines, etc.

Using many of these primitives and predicates, solutions to several problems have been extended to $\mathbb{T}^2$, some of which are illustrated on the accompanying video. Among them:

*Supported by a fellowship from Capes/Brazil.
†Supported by a fellowship from CNPq/Brazil.

SCG'05, June 6–8, 2005, Pisa, Italy.
ACM 1-58113-991-8/05/0006.

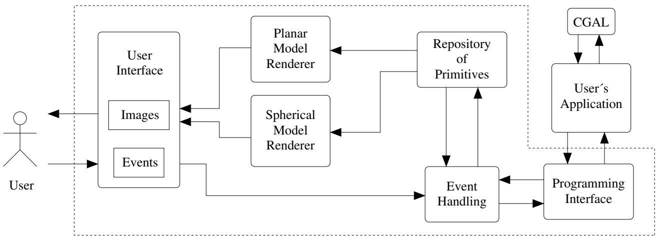
construction of the convex hull of sets of points, arbitrary and Delaunay triangulation (covering the entire space and not just one range), Voronoi diagram (closest and farthest neighbor), minimum distance spanning tree, largest empty circle (with proper center), all nearest neighbor graph, point location on $\mathbb{T}^2$ subdivisions, detection and enumeration of intersections of segments (of finite and infinite length) and range search with isothetic rectangles.

3. $\mathbb{T}^2$VIEWER

In order to benefit from the extension of CGAL to the oriented projective plane in the context of the classroom, the need for visualization arose. So, we developed a dynamic visualization system, $\mathbb{T}^2$Viewer [7], which is also illustrated in the video that accompanies this paper. $\mathbb{T}^2$Viewer can display the planar as well as the spherical models of the OPP, and allows for interaction with extended CGAL applications, providing means for output display and input generation — user created elements as well as user selection of objects previously generated by other applications.

$\mathbb{T}^2$Viewer was developed in C++, following CGAL's programming guidelines [3], relies on OpenGL for fast dynamic visualization of the three dimensional models and uses Qt-Designer as its framework. Similar to modifications done to the *kernel* and the *basic_library* of CGAL for its extension to the OPP, the development of $\mathbb{T}^2$Viewer required modifications to CGAL's *support_library*.

The following figure shows a high level diagram of the conceptual modules that comprise $\mathbb{T}^2$Viewer.



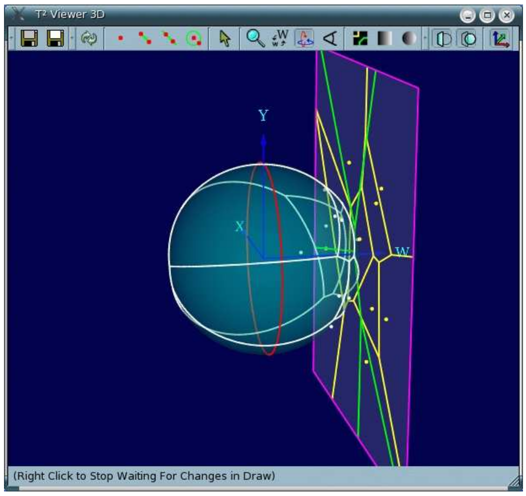
Some of the main characteristics of $\mathbb{T}^2$Viewer are:

- simultaneous display of both models which allows the viewer the benefit of choosing the most convenient one in each given situation;
- opacity adjustment for either model, allowing for the visualization of one side only, or both at once;
- zoom out and zoom in (all the way to the center of the spherical model — the origin of the projections);
- broadening and narrowing of the field of view;
- scaling the homogeneous coordinate which permits focusing on details near the origin or on the behavior of objects near infinity;
- some ability to produce algorithm animation.

4. THE VIDEO

The accompanying video was captured in realtime (with minimum subsequent edition) to show a few algorithms implemented on the oriented projective plane under extended CGAL in action and some of the features of $\mathbb{T}^2$Viewer. We chose to illustrate user interaction with both models of $\mathbb{T}^2$ for input generation (triangulation), query actions (point location), visualization of either or both models (in 3-D orbiting), smooth variation of surface transparency to show either or both ranges of $\mathbb{T}^2$, algorithm animation (segments intersections) and the use of input tools. Furthermore, while displaying the interactive construction of the Voronoi diagram, the video illustrates the duality provided by the OPP, which allows for the simultaneous construction of the order 1 (nearest neighbor) diagram and the order $n-1$ (farthest neighbor) diagram of points on the front range.

The following figure illustrates one of the views provided by $\mathbb{T}^2$Viewer. See www.ic.unicamp.br/~rezende/T2Viewer



5. ACKNOWLEDGMENTS

We thank K.A.R. for lending her voice for the audio track.

6. REFERENCES

[1] P. J. de Rezende and J. Stolfi. *Fundamentos de Geometria Computacional*. UFPE-DI, IV Escola de Computação, 1994.

[2] P. J. de Rezende and R. B. Westrupp. An Optimal Algorithm to Construct All Voronoi Diagrams for k Nearest Neighbor Search in $\mathbb{T}^2$. *In Proceedings of the XII SIBGRAPI*, pages 7–15, 1999.

[3] CGAL developers' manual, November 2002. http://www.cgal.org

[4] C. N. Gon. Computação Exata em Geometria Projetiva Orientada e Tratamento de Degenerações. Master's thesis, UNICAMP, 1996.

[5] A. G. Oliveira. Implementação do Plano Projetivo Orientado na Biblioteca CGAL. Master's thesis, UNICAMP, 2004.

[6] G. A. Pinto. Generalizações do Diagrama de Voronoi construídas através de Cônicas no Plano Projetivo Orientado. Master's thesis, UNICAMP, 1998.

[7] F. P. Selmi-Dei. Um Visualizador para uma Extensão de CGAL ao Plano Projetivo Orientado. Master's thesis, UNICAMP, 2005.

[8] J. Stolfi. *Oriented Projective Geometry: A Framework for Geometric Computations*. Acad. Press, Inc., 1991.

[9] R. B. Westrupp. Diagramas de Voronoi de Ordem k na Geometria Projetiva Orientada. Master's thesis, UNICAMP, 1999.