Pseudo-Triangulation Workbench (ptw)
User Manual

Lutz Kettner*

Revision: 1.2. Date: 2001/09/03

1 Introduction

The Pseudo-Triangulation Workbench (ptw) is a research tool to investigate pseudo-triangulations of points in the plane. It allows to create point sets, compute pseudo-triangulations using various algorithms, interact with them using flips, highlight interesting properties using different color schemes, and enumerate all pseudo-triangulations for a given point set. The enumeration can also be used with a branch&bound version to test hypothesis, such as minimal vertex degree. Currently, the algorithms and enumeration for pseudo-triangulations create only minimal pseudo-triangulations. Some support exists for normal triangulations, such as creation and diagonal flips.

I wrote most of this program during the January 2001 Workshop on Pseudo-Triangulations in Barbados. I spend around 25 hours during the workshop and another 25 hours after the workshop on implementing and documenting this prototype. It is based on my halfedge data-structure released with CGAL 2.3[1]. It uses CGAL for the geometry, OpenGL for the graphics, and GLUT with GLUI for the window GUI surface.

Later, I developed the additional command-line tools for enumerating pseudo-triangulations and applied them to the data base of all order types of small point sets in the plane to up to ten points, see [1]. I was able to compute the number of pseudo-triangulations for up to and including nine points. Ten points are expected but not ready yet.

2 Home Page and Feedback

The home page is `http://www.cs.unc.edu/~kettner/proj/PseudoT/`. Comments, suggestions for extensions, and bug-reports are welcome. Please write me at `kettner@cs.unc.edu`.

*`kettner@cs.unc.edu`

[1] `www.cgal.org`

3 DISCLAIMER

This program is a research prototype! It does not provide a clean, homogeneous or foolproof user interfaces. It is easy to bring it in an inconsistent state where answers of the system are plain meaningless. However, I have not seen it crashing a lot. ;-)

The underlying geometry is not robust! It uses currently machine double arithmetic (could be changed easily but there is no need as long as we work with input points distinguishable in the graphics window). It also does not handle degeneracies, e.g., three points on a line. I have not spend the time to have something for the definition of *reflex vertex* in that case.

The enumeration is supposed to enumerate all pseudo-triangulations for a given set of points. Correctness of the implementation has been validated with some example cases of wheels where we do know the exact number. More confidence has been established with a second, independent implementation of the same algorithm by Hervé Brönnimann that gives the same answers for all point sets in the order type data base for up to and including nine points.

Not scalable! This program is designed for interactive exploration. Especially the interactivity and visualization are quite wasteful in time assuming there will be only small to middle sized pseudo-triangulations. However, several of the algorithms, the flip, and especially the enumeration are efficient, and the enumeration has undergone a first tuning.

4 Availability and Installation

The source code and the user manual are available for download from the home page at `http://www.cs.unc.edu/~kettner/proj/PseudoT/`. See the README file and the INSTALL file for installation instructions and system requirements.

5 View Transformation

The program distinguishes between image (i.e. world) space and object space, where the pseudo-triangulation lives. The affine transformation between the two has a translation and a scale component. The cursor keys `[Up]` `[Down]` `[Left]` `[Right]` translate in the corresponding direction, the keys `[PgUp]` `[PgDown]` scale up and down. `[Reset View]` initializes the transformation to its initial setting.

6 Mouse Events and Selected Items

A left mouse click selects a point, edge, or face. A selection of a point has priority over a selection of an edge has priority over a selection of a face. Points and edges are expanded with a ε-neighborhood to make their selection easier. Faces are selected using vertical ray shooting simultaneously selecting the edge above the point. If there is no edge above the point the infinite outer face and some arbitrary edge on the convex hull is selected.

The button down event selects. For each mouse move the selection is evaluated again, or in the case of a selected point the point is dragged along. The button up event makes the selection final.

Directly clicking on an edge selects the edge, flips it, and de-selects it.

The `[Esc]` key de-selects all selected items.

A right mouse click creates a new point where is none, and deletes a point where is one.

7 Creation, Algorithms, and Flips

The user interface allows to create points by right-clicks with the mouse or using generators. A generator creates as many points as entered in the `''#Pts''` entry box. A choice box gives access to different generators listed in the following table. The generator is executed with the `[Create]` button or the `[P]` shortcut key. Points are created in image space. Thus translating and scaling allow the composition of differently generated point sets. `[Clear]` deletes all points.

Rnd in Disc	random points uniformly chosen in a disc
Reg on circle	regularly spaced points on a circle
Reg wheel	regularly spaced points on a circle plus center (wheel)

Given a point set, algorithms can be invoked to compute a pseudo-triangulation or triangulation. Just selecting an algorithm from the choice box already calls it. The hot key `[Enter]` calls it as well. Since the user interface allows dragging of points, a continuous update could be nice. The hot key `[D]` toggles the dynamic flag. It is off by default. If it is on, the selected algorithm is called after each action at the user interface including mouse clicks and moves. (Hint: If you click an edge to flip it and it does not flip, this dynamic flag might still be active and just recompute the original triangulation after the flip has been done, effectively canceling out the flip. Hitting `[D]` leaves a message in the status line identifying the state of this flag.) The following algorithms are currently available.

LR Triang
RL Triang
LR Pseudo
RL Pseudo
Greedy Pseudo

LR algorithms work on a sorted list of points from left to right, RL from right to left. This type of algorithm just connects each new point to the convex hull of the already processed set of points. Triang computes a triangulation creating a fan from the new point to the convex hull. Pseudo creates a minimal pseudo-triangulation connecting only the two new tangents. Greedy Pseudo implements the greedy pseudo-triangulation of Michel Pocchiola starting with a downwards direction and rotating it counterclockwise creating new bi-tangents on the way (implemented differently here). This pseudo-triangulation is also used as the starting point for the enumeration algorithm. The vertical direction is a degenerate case (stop condition for the enumeration). As a solution, I consider the point set as if infinitesimally rotated clockwise.

`[Clear Edges]` deletes all edges leaving only the points.

Given a triangulation or pseudo-triangulation, diagonals can be flipped by clicking on them. Diagonals can always be flipped in a pseudo-triangulation, but I forbid flips that would result in a disappearing diagonal, i.e., a new diagonal identical to an existing edge. Using such a flip criterion works also nicely as a flip in a normal triangulation. The user interface remembers the most recently flipped edge. The hot key `[F]` flips that edge again, which allows to flip a diagonal conveniently back and forth several times.

8 Validity Check

I have a validity test that checks data structure consisistency, minimality of the pseudo-triangulation and proper angles. `[v]` calls the validity check and reports the result in the status line at the bottom of the window. `[V]` gives in addition a detailed report on the tests on the standard error output. This report is rather specialized and probably only useful for debugging.

9 Shadow

The user interface allows to create a *shadow* from a pseudo-triangulation `[Shadow]` or `[S]`, restore the pseudo-triangulation from a shadow `[Restore Shadow]`, or clear the shadow `[Clear Shadow]`. The edges of the shadow are shown in a lighter color behind the current pseudo-triangulation. The shadow can be useful to study the overlay of two pseudo-triangulations and it also used to memorize a result of the enumeration.

The shadow is no longer accessible with the mouse nor is it connected to the point set. Dragging a point does not alter the shadow.

10 Color Schemes, Selected Items, and Circumcircle

Three color schemes can be selected: one for vertices, one for edges, and one for faces. A vertex color scheme can be chosen from the following list:

<red>	default is a red disc with a black circle around it.
Reflex	proper reflex vertices are red, others are white.
Degree	vertex degree: 2 = yellow, 3 = green, 4 = red, 5 = pink, everything else black.

An edge color scheme can be chosen from the following list. The data structure has a *marked* flag for edges. It is currently used in the enumeration to mark *fixed* (or *red*) edges. However, when a pseudo-triangulation gets reported usually all edges are marked except convex hull edges.

<black>	default is a black line.
Marked	*marked* edges blue, others black.
Flippable	flippable edges blue with a darker blue for those where both diagonals share a common endpoint. Others are black.
Imprv.Lift	edges whose flipped diagonal improves in the lifting map are green. Flippable edges with a common endpoint for both diagonals are darker blue (they don't improve in the lifting either way). Others are black.
Imprv.Angle	edges whose flipped diagonal increases the smallest angle are green. The edge whose flip increases the smallest of all angles is in a brighter green. Others are black.

Selected items are shown in yellow. For a selected face its surrounding edges are drawn in a darker yellow, the selected edge in a lighter yellow. A selected point is drawn in a kind of orange yellow.

The only available face color scheme draws the circumcircle through the three corners of the selected pseudo-triangle (no circumcircle for the outer face).

11 Enumerating

The enumeration uses Michel Pocchiola's efficient algorithm. The enumeration needs to be initialized, `[Init]` or `[I]`. Then, one can step through all pseudo-triangulations one by one, `[Step]` or `[Space]`, or run through them continuously – either drawing each one, `[Animate]`, or just creating and counting them, `[Run]` or `[R]`.

Once the enumeration is running continuously it cannot be interrupted. The status line at the bottom gives progress reports such as numbers counted and timing. The result, e.g., number of pseudo-triangulations, is printed in the status line as well as on standard error output, where it remains accessable for further use.

The enumeration can be used for searching specific species of pseudo-triangulations. If one is found it is stored as a shadow to report it. The following searches have been implemented so far. Branch&bound is used to cut branches of the search trees wherever possible.

Count Pseudo	counting all pseudo-triangulations.
Min-max-degree	searching for a pseudo-triangulation with a minimum maximal vertex degree.
Min-max-6	searching for a pseudo-triangulation with a minimum maximal vertex degree of five or less. A negative answer would give a lower bound for the constant vertex degree question.

12 Keyboard Shortcuts

Except for `[v]` and `[V]`, lower and uppercase letters are equally accepted.

Group	Key	Function
General	`[Esc]`	de-select all selected items.
	`[Ctrl][Q]`	quit.
	`[P]`	create point set using selected generator.
	`[C]`	clear point set.
	`[S]`	create shadow.
	`[v]`	validate current pseudo-triangulation. Print result in status line.
	`[V]`	validate current pseudo-triangulation. Print result in status line and long report to standard error output.
View	`[Up][Down]`	translate viewing area.
	`[Left][Right]`	translate viewing area.
	`[PgUp][PgDown]`	scale viewing area.
Algo	`[Enter]`	executes the currently selected algorithm.
	`[D]`	toggle dynamic algorithm tag. If on, the currently selected algorithm runs after each mouse move or click.
	`[F]`	flip back the most recent edge flipped.
Enum	`[I]`	initialize enumeration.
	`[Space]`	one step in the enumeration.
	`[R]`	run enumeration.

13 Missing Features and Future Plans

File I/O is missing for the pseudo-triangulation, so there is no possebility to save or load a configuration right now.

New color schemes and new tests for enumerations are easy to add. Suggestions are welcome.

References

[1] Oswin Aichholzer, Franz Aurenhammer, and Hannes Krasser. Enumerating order types for small point sets with applications. In *Proc. 17th Ann. ACM Symp. Computational Geometry*, Medford, Massachusetts, USA, 2001. `http://www.igi.TUGraz.at/oaich/triangulations/ordertypes.html`.