In-Core and Out-Core Memory Fast Parallel Triangulation Algorithm for Large Data Sets in E² and E³

Michal Smolik[1] Vaclav Skala[2]
University of West Bohemia, Plzen, Czech Republic

1 Introduction

Today's applications need to process large data sets using several processors with a shared memory, i.e. in parallel processing, or/and on systems using distributed processing. In this paper we describe an approach applicable for effective triangulation in E^2 and E^3 (tetrahedralization) for large data sets using CPU and/or GPU parallel or distributed systems, e.g. computational clusters.

In many cases we do not need exact Delaunay triangulation [Chen 2011] or another specific triangulation. Triangulation as "close enough" to the required type of triangulation is acceptable. Weakening this strict requirement enables us to formulate a simple "Divide & Conquer" algorithm [Cignoni et al. 1998]. The approach is independent from the triangulation property requirements.

2 Principle of the Proposed Algorithm

The given data set can be split to several subsets, not necessarily rectangular, i.e. to $n \times m$ domains in E^2, resp. $n \times m \times p$ in E^3. Each data subset contains the original points plus additional corner points of the appropriate domain. Every domain is triangulated using any triangulation library.

Now, joining two triangulated domains is simple as those two domains share the same corner points. We only have to replace the common edge EF by the edge AB (see Fig. 1). It can be seen that the connection of triangulated subsets is extremely simple in the E^2 case. In the E^3 case the situation is similar and simple too.

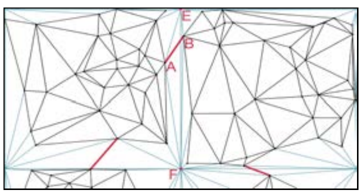
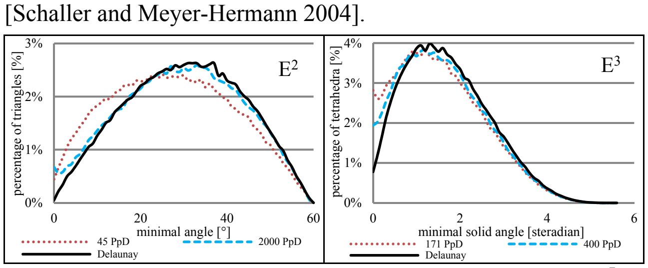


Figure 1: *Joining triangulated domains by edge EF→AB swapping.*

The corner points can be retained in the tri-angulation, or can be re-moved and "star-shape" holes have to be re-tri-angulated [Schaller and Meyer-Hermann 2004].



Graph 1&2: *Distribution of minimal internal angles for 10^7 uniformly distributed points (PpD = Points per Domain).*

[1]e-mail: smolik@kiv.zcu.cz, [2]web: www.vaclavskala.eu

SIGGRAPH 2014, August 10 – 14, 2014, Vancouver, British Columbia, Canada.

ACM 978-1-4503-2958-3/14/08

It should be noted that domain triangulations are totally independent. If the domains' corner points have to be removed, the created holes have to be tessellated. The processes are independent again and can be executed totally in parallel.

In the case of large data, we do not have to process all domains at once, but can process them in some smaller parts considering the size of available memory. As domain triangulation and their joining are independent, there is no change in the proposed algorithm and it can be implemented easily.

3 Experimental Results

The quality of the triangulation has been tested on data sets with uniform distribution. The Delaunay triangulation maximizes the minimum angle therefore it is appropriate to test the distribution of minimal internal angles in triangles, resp. tetrahedra, see Graph 1&2. The more points per domain that are used, the closer our triangulation is to the Delaunay triangulation.

The Delaunay triangulation maximizes the mean 'inradius'. It can be seen that, in the case of removing the corner points, there is only a 5% difference to the Delaunay triangulation.

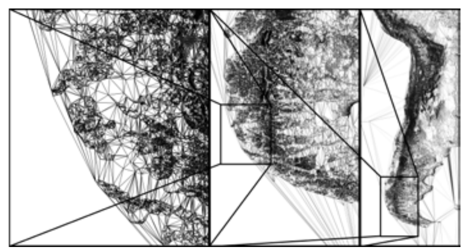


Figure 2: *Triangulation of South America.*

The approach proposed has been tested on synthetic & real data sets, e.g. South Americas GIS data set, Fig. 2. Running time for $1.1 \cdot 10^6$ points was $0.42[s]$, on uniform data set in E^2 the running time for 10^8 points was $28.2[s]$ and for 10^6 points enables real time performance using large number of cores. In E^3 runtime was $25.7[s]$ for 10^7 points, on PC Core i7 (4×2.67GHz), 12 GB.

4 Conclusions & Acknowledgments

A new fast parallel triangulation algorithm in E^2 and E^3 has been implemented on parallel environments with shared and/or distributed memory using both CPU and GPU. As it is scalable, the proposed algorithm is especially convenient for processing large data sets. The proposed approach has been implemented and tested using CPU and GPU as well. Research was supported by MSMT CR LG13047, LH12181 and SGS 2013-029.

References

CHEN, M.-B. 2011. A Parallel 3D Delaunay Triangulation Method, *9th ISPA 2011*, IEEE, pp.52-56.

CIGNONI, P., MONTANI, C., AND SCOPIGNO, R. 1998. DeWall: A Fast Divide & Conquer Delaunay Triangulation Algorithm in Ed, *Computer Aided Design*, Vol.30, No.5, pp.333-341.

SCHALLER, G., AND MEYER-HERMANN, M. 2004. Kinetic and Dynamic Delaunay Tetrahedralization in Three Dimensions, *Computer Physics Communications*, Vol.162, No.1, pp.9-23.