

Visual Data Mining Using the Point Distribution Tensor

Marcel Ritter

Graduate School for Scientific Computing
University of Innsbruck
Innsbruck, Austria
marcel.ritter@uibk.ac.at

Werner Benger

Center for Computation & Technology
Louisiana State University
Baton Rouge, USA
werner@cct.lsu.edu
Institute for Astro- and Particle Physics
University of Innsbruck
Innsbruck, Austria
werner.benger@uibk.ac.at

Biagio Cosenza

Distributed and Parallel Systems Group
University of Innsbruck
Innsbruck, Austria
cosenza@dps.uibk.ac.at

Keera Pullman

ESRI Australia
Darwin, NT, Australia
kpullman@esriaustralia.com.au

Hans Moritsch

Distributed and Parallel Systems Group
University of Innsbruck
Innsbruck, Austria
hans@dps.uibk.ac.at

Wolfgang Leimer

Distributed and Parallel Systems Group
University of Innsbruck
Innsbruck, Austria
wolfgang.leimer@student.uibk.ac.at

Abstract—We explore a novel algorithm to analyze arbitrary distributions of 3D-points. Using a direct tensor field visualization technique allows to easily identify regions of linear, planar or isotropic structure. This approach is very suitable for visual data mining and exemplified upon geoscience applications. It allows to distinguish, for example, power lines and flat terrains in LIDAR scans. We furthermore present the work on the optimization of the computationally intensive algorithm using OpenCL and potentially utilizing the Insieme optimizing compiler framework.

Keywords—metric tensor; scientific visualization; point cloud; OpenCL.

I. INTRODUCTION

Point clouds occur as primary data sources in different scientific domains, e.g., stemming from simulations in computational fluid dynamics by smooth particle simulations or from observational methods, such as light detection and ranging (LIDAR) laser scanning [1]. Classification of point clouds is still ongoing research for LIDAR laser scan data [2]. Geometric information about the local point distribution can be used for classification, for constructing surfaces, or as basis for other algorithms. An algorithm to compute Gaussian and mean curvature on polygon meshes was presented in [3], based on the tensorial product of the polygon’s normal vectors. A product with additional weights was used to compute the co-variance matrix of point neighborhoods describing tangential frames for surfaces in [4]. This co-variance matrix provides us with a type of smooth transition between lines, surfaces, and volumes [5].

In this article, we utilize the direct tensor visualization technique [6] to illustrate the co-variance matrix resulting from arbitrary point clouds. Section II introduces the

distribution tensor and the utilized visualization technique, presented on simple geometric point distributions. Two algorithms for the tensor computation are described: One for central processing units (CPUs) and one for graphics processing units (GPUs). Optimizations are presented and the Insieme compiler optimization framework [7] is introduced. Our visualization method is demonstrated on two geo-scientific applications in Section III: On the analysis of LIDAR laser scan data and the analysis of coastlines. The paper concludes and describes future work in Section IV.

II. COMPUTING THE POINT DISTRIBUTION TENSOR

A. Mathematical Background

We define the “point distribution tensor” as a measure constructed from a set of N points $\{P_i : i = 1 \dots N\}$ similar to the co-variance matrices in [4] [8] [5]:

$$S(P_i) = \frac{1}{N} \sum_{k=1}^N \omega_{ik}(t_{ik} \otimes t_{ik}^r) \quad (1)$$

whereby $t_{ik} = P_i - P_k$ and an optional weighting function $\omega_{ik} := f(\|P_i - P_k\|, r, i)$. Here, r is an user specified distance or radius defining the neighborhood of point P_i . The weighting function ω_{ik} is zero outside this radius. The distribution tensor is symmetric and positive definite such as the metric tensor [9] and, thus, yields three eigen-values when doing an eigen-analysis: $\lambda_3 \geq \lambda_2 \geq \lambda_1$. These are used to classify the tensor via three shape factors [10], characterizing the shape of a fitting ellipsoid of the point

neighborhood in barycentric coordinates, see Figure 1:

$$\begin{aligned} c_{linear} &= (\lambda_3 - \lambda_2) / (\lambda_1 + \lambda_2 + \lambda_3) \\ c_{planar} &= 2(\lambda_2 - \lambda_1) / (\lambda_1 + \lambda_2 + \lambda_3) \\ c_{spherical} &= 3\lambda_1 / (\lambda_1 + \lambda_2 + \lambda_3) \end{aligned} \quad (2)$$

with $c_{linear} + c_{planar} + c_{spherical} = 1$. A tensor field visualization method more suitable for large data than drawing tensor ellipsoids is utilized. Instead of ellipsoids textured splats are rendered with smooth transitions in color, orientation, texture and transparency, as shown in Figure 1.

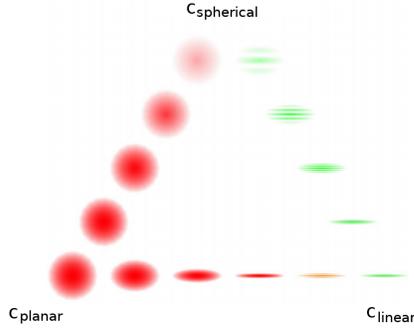


Figure 1. Tensors are visualized as textured oriented disks. The three shape factors, Equation 2, are used for smooth transitions between linear (right), planar (left), and spherical (top) shape [9]. In this context, the disks enhance the visualization of points predominantly distributed on a line, on a surface, or in a volumetric distribution.

B. Test Cases

Simple analytic test cases were used to verify and study the properties of the distribution tensor, as illustrated in Figure 2. The point distributions have an extent of 1.0 in spatial dimensions and have been computed using a neighborhood radius of $r = 0.2$. Figure 2 (a) shows linear tensor splats textured and oriented in one direction. At the corners of the rectangle tensors become planar caused by two equally dominant directions in the neighborhood. Homogeneous distributions are fully transparent and become invisible, as demonstrated in Figure 2 (c). Here, the inner region is transparent, the border surfaces become more planar and are colored red while corner points become linear (green).

C. Algorithm

A first serial algorithm was implemented in the visualization shell VISH [11] utilizing C++ and OpenGL. Computation and visualization tasks were split in different modules. The computation module searches for neighbors in a 3D KD-Tree [12] within a user specified radius (where $\omega_{ik} > 0.0$) to limit the number of considered points. Alternatively to setting the radius, also the number of neighbors can be specified. Furthermore, a scalar field given on the points can be utilized to set the radius or number of points for each point individually. Eqn. 1 is utilized to compute the distribution tensor for each point. Different weighting

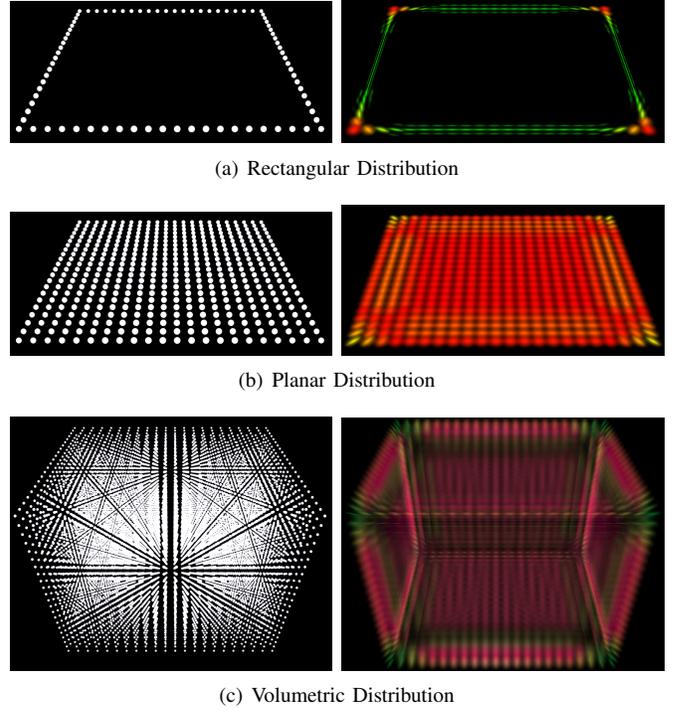


Figure 2. *Left*: Analytic point distributions illustrated by simple point rendering. *Right*: Corresponding distribution tensor fields. Linear 1D, planar 2D and isotropic 3D tensors are visualized using tensor splats, having a dominant linear, planar and spherical shape factor, respectively.

functions have been implemented inside the neighborhood: $\omega_{ik} := (r - \|P_i - P_k\|)/r$, $\omega_{ik} := 1/r$ and $\omega_{ik} := 1/r^2$.

Data is represented in an unified data model [9] [13] which allows support of different types of grid geometries and topologies. The computation algorithm operates on the vertices of any grid type. In the following applications point clouds and sets of lines are used for analysis.

D. GPU implementation

An alternative implementation of the algorithm was done in OpenCL [14], a framework for multicores and parallel hardware being able to execute programs also across heterogeneous platforms. The neighborhood is controlled by a fixed radius. Instead of the KD-Tree, a uniform grid was preferred as data structure to speed-up the neighborhood search. Here, the *loose grid* approach was adapted, where each particle is assigned to one cell based on its position. The grid's cell size depends on the influence radius (i.e., $\text{radius} \geq \text{cell size}$). Therefore, each particle can affect the closest 27 cells while calculating the tensor. This method allowed to bin the particles into the cells and to sort them by their grid index. The algorithm comprises four steps:

- 1) for each particle a hash value is computed, i.e., the cell index where it is located;
- 2) particles are sorted by hash; for this step NVidia's optimized bitonic sorting [15] is utilized;

- 3) the sorted list is used to compute the starting cell where the particle is located, running a thread for each particle, and performing scattered memory writes;
- 4) tensor calculation: Each particle searches the closest 27 grid cells from its location and it computes the tensor with each of the particles in these cells.

Steps 1 – 3 are related to the build process of the grid data structure. The sorting algorithm is highly effective because it improves the memory access coherency when calculating the tensor, and reduces thread divergence (particles in the same thread group tend to be close together in space).

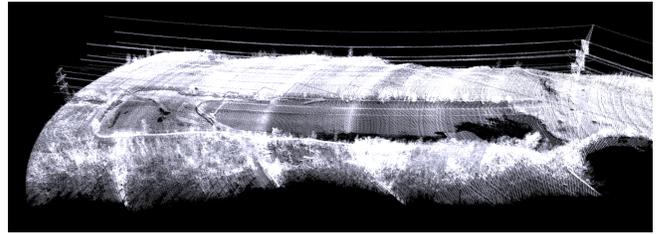
E. Optimization

The CPU algorithm was parallelized using OpenMP [16], adding a minimal overhead in development. Furthermore, OpenMP, as also OpenCL is supported by Insieme [7].

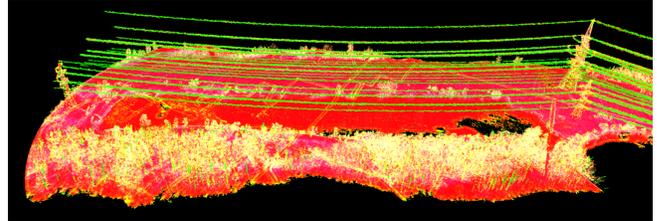
The Insieme compiler, under development at the University of Innsbruck, is a source-to-source compiler for C/C++ aiming at the automatic optimization of parallel programs implemented with MPI, OpenMP or OpenCL. It optimizes the source code for a specific platform (e.g., NVidia Fermi architecture), and applies transformations such as loop enrolling and collapsing, thread merge and data pre-fetching. Insieme aims at supporting programmers in effectively optimizing programs across different architectures, including shifting of computations from CPU to GPU cores. Optimizations are performed at compile-time through code analysis and transformations for sequential and parallel code regions. An intermediate representation is facilitated which explicitly describes parallelism, synchronization, and communication. The program's behavior is optimized and customized to the available hardware resources at runtime by utilizing statistical machine learning techniques based on a performance analysis database. Performance measures are, e.g., execution time, energy consumptions, and computing costs. Preliminary tests will be done with this code which is now part of the Insieme test cases.

III. APPLICATION RESULTS

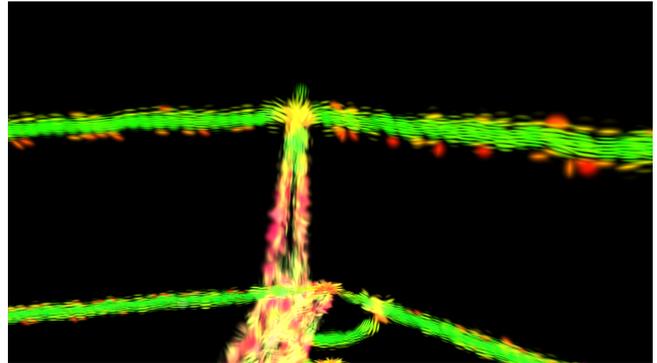
The method was applied to two different geoscience applications. Figure 3 shows a scan of a water basin close to the Danube in Austria captured with the Riegler hydrographic laser scanner VQ-820G [17]. In the example, a fixed neighborhood radius of two meters and a constant weighting function $\omega_{ik} = 1$ showed good results. Other parameters for r and ω_{ik} have been tested as well. Figure 3 (a) illustrates the received and processed laser echoes as a cloud of points; (b) and (c) show the distribution tensor field. Linear structures, such as the power cables, are well identified (green). The ground is dominantly planar (red). Some regions of the ground fade to magenta indicating less planarity. Here, grass influences the planar tensor to become more isotropic. Bushes and trees are isotropic or of an interpolated intermediate shape, mostly appearing



(a) LIDAR Echos



(b) LIDAR Tensors $r = 2.0m$



(c) LIDAR Tensors $r = 2.0m$ Detail

Figure 3. Distribution tensor field of returned laser echoes from an airborne laser scan. Linear distributions such as cables and planar distributions such as ground are emphasized. Vegetation is fading to spherical.

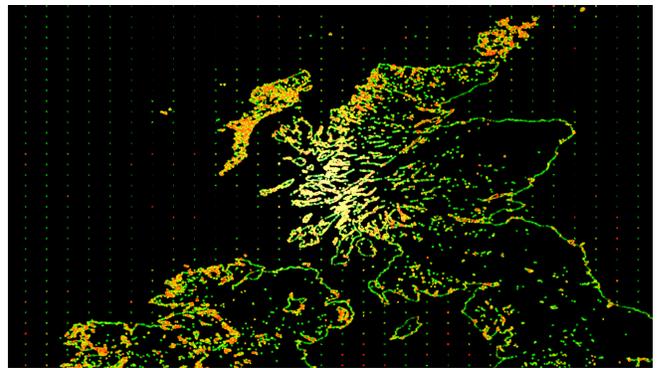


Figure 4. Distribution tensor field of an ESRI shapefile of the earth's water bodies and coastlines. The distribution tensor field with 12 fixed neighbors of the northern part of the United Kingdom is illustrated.

yellow. The computation with the OpenMP version utilizing 4 threads of the 4.81mio points with approximately 600 neighbors per point ($r = 2.0m$) took 752 seconds on a i7 M640 2.8GHz with 7.7GB RAM and NVidia Quadro FX3800M using Linux64bit, gcc 4.4.5, and Vish SVN 3854.

Another application was the analysis of coast and contour lines. Shapefiles [18] of water bodies and coastlines were investigated. Figure 4 (b) shows the distribution tensor field of the coast of the United Kingdom. Unstructured coastlines are highlighted in green whereas cliffy coast lines are shown in red, for example when looking at the northern coast of Scotland. Here, a rather small neighborhood of fixed 12 points turned out to emphasize cliffy coasts.

IV. CONCLUSION AND FUTURE WORK

A new method of enhancing the visualization of point distributions was introduced, described, and demonstrated. Two different implementations and parallelization approaches were presented. Using an unified data model opened the possibility to apply the technique to data sets stemming from two different scientific applications: The visual extraction of power cables in LIDAR data and the visual enhancement of cliffy coastlines. We will further use the tensor analysis on LIDAR data to enhance point classification and the creation of digital terrain models. Different weighting functions and parameter studies will be investigated on more datasets. We ultimately will use the Insieme framework to optimize our parallel GPU and OpenMP codes.

ACKNOWLEDGMENT

Thanks to Frank Steinbacher for providing the LIDAR data sets. This work was supported by the Austrian Research Promotion Agency (FFG) *Airborne Hydromapping*, the Austrian Science Foundation FWF DK+ project *Computational Interdisciplinary Modeling (W1227)*, and grant P19300. This research employed resources of the Center for Computation and Technology at Louisiana State University, which is supported by funding from the Louisiana legislatures Information Technology Initiative. This work was supported by the Austrian Ministry of Science BMWF as part of the UniInfrastrukturprogramm of the Forschungsplattform Scientific Computing at LFU Innsbruck.

REFERENCES

- [1] E. P. Baltsavias, "Airborne laser scanning: existing systems and firms and other resources," *ISPRS Journal of Photogrammetry & Remote Sensing*, vol. 54, pp. 164–198, 1999.
- [2] P. Dorninger, B. S. A. Zamolyi, and A. Roncat, "Automated Detection and Interpretation of Geomorphic Features in LIDAR Point Clouds," no. 99, pp. 60–69, 2011.
- [3] G. Taubin, "Estimating the tensor of curvature of a surface from a polyhedral approximation," in *Proceedings of the Fifth International Conference on Computer Vision*, ser. ICCV '95. Washington, DC, USA: IEEE Computer Society, 1995, pp. 902–.
- [4] M. Alexa, S. Rusinkiewicz, M. Alexa, and A. Adamson, "On normals and projection operators for surfaces defined by point sets," in *In Eurographics Symp. on Point-Based Graphics*, 2004, pp. 149–155.
- [5] J. Berkmann and T. Caelli, "Computation of surface geometry and segmentation using covariance techniques," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 16, no. 11, pp. 1114 –1116, nov 1994.
- [6] W. Benger and H.-C. Hege, "Tensor splats," in *Conference on Visualization and Data Analysis 2004*, vol. 5295. Proceedings of SPIE Vol. #5295, 2004, pp. 151–162.
- [7] DPS Group at Universität Innsbruck, "The insieme compiler project." [Online]. Available: <http://www.dps.uibk.ac.at/insieme/>
- [8] A. Adamson, "Computing curves and surfaces from points," Ph.D. dissertation, TU Darmstadt, 2008.
- [9] W. Benger, "Visualization of general relativistic tensor fields via a fiber bundle data model," Ph.D. dissertation, FU Berlin, 2004.
- [10] C. Westin, S. Peled, H. Gudbjartsson, R. Kikinis, and F. Jolesz, "Geometrical diffusion measures for mri from tensor basis analysis," in *Proceedings of ISMRM, Fifth Meeting, Vancouver, Canada*, Apr. 1997, p. 1742.
- [11] W. Benger, G. Ritter, and R. Heinzl, "The Concepts of VISH," in *4th High-End Visualization Workshop, Obergurgl, Tyrol, Austria, June 18-21, 2007*. Berlin, Lehmanns Media-LOB.de, 2007, pp. 26–39.
- [12] J. H. Friedman, J. L. Bentley, and R. A. Finkel, "An algorithm for finding best matches in logarithmic expected time," *ACM Transactions on Mathematics Software*, vol. 3, no. 3, pp. 209–226, September 1977.
- [13] M. Ritter, "Introduction to HDF5 and F5," Center for Computation and Technology, Louisiana State University, Tech. Rep. CCT-TR-2009-13, 2009.
- [14] KHROSOS Group, "OpenCL," 2011. [Online]. Available: <http://www.khronos.org/opencv>
- [15] K. E. Batcher, "Sorting networks and their applications," in *Proceedings of the April 30–May 2, 1968, spring joint computer conference*, ser. AFIPS '68 (Spring). New York, NY, USA: ACM, 1968, pp. 307–314.
- [16] OpenMP Architecture Review Board, "OpenMP," 2011. [Online]. Available: <http://openmp.org>
- [17] F. Steinbacher, M. Pfennigbauer, A. Ulrich, and M. Aufleger, "Vermessung der Gewässersohle - aus der Luft - durch das Wasser," in *Wasserbau in Bewegung ... von der Statik zur Dynamik. Beitrge zum 15. Gemeinschaftssymposium der Wasserbau Institute TU München, TU Graz und ETH Zürich*, 2010.
- [18] ESRI, "ESRI Shapefile Technical Description," Environmental Systems Research Institute, Inc, White Paper, July 1998.