Project number IST-25582

CGL

Computational Geometric Learning

D2.3: Work Package 2 [Period 2] Report

STREP

Information Society Technologies

Period covered:	November 1, 2010–October 31, 2011
Date of preparation:	October 30, 2012
Date of revision:	
Start date of project:	November 1, 2010
Duration:	3 years
Project coordinator name:	Joachim Giesen (FSU)
Project coordinator organisation:	Friedrich-Schiller-Universität Jena Jena, Germany

In the following we describe the work done within Work Package 2 within the second period. In the description we follow the structure imposed by the tasks for this work package and period. Wherever there is no deviation and all goals have been met, it is *not* mentioned specifically. We start by restating the objectives for Work Package 2 and conclude with a discussion of the milestones for Period 1.

Objectives

Computational Geometry has been very successful in the past 25 years in developing methods for problems in low dimensions, and CGAL has managed to provide a unified software framework for implementing such methods.

Within this work package, fundamental *high-dimensional* data structures and algorithms will be developed, in most cases with the goal of implementing them in CGAL. High-dimensional geometric data processing is now in a situation that we have encountered fifteen years ago in the low-dimensional setting: a number of useful theoretical concepts has been developed over the past years, but robust and efficient implementations are mostly lacking. In order to have an impact in practice, such implementations must scale with the dimension and exploit (hidden) structure of the data.

In analyzing high-dimensional data structures and algorithms, our focus is not on worst-case complexity, but on (provably) good performance under some given structural properties of the input. These properties may be of statistical nature (when we are dealing with noise, for example), or of geometric nature (when data is of low intrinsic dimension, say). Related to this, we are also aiming at output-sensitive algorithms.

Tasks

Task 1.a: Derivative free methods

The *Random Pursuit* (RP) algorithm for minimizing a convex function samples in each iteration a direction from the unit sphere and then performs a line search for the optimum point in this direction to obtain the next iterate. We have developed and successfully analyzed this algorithm in Period 1 [18, 20]. The main drawback of RP is that it is non-adaptive, i.e. it does not learn a sampling distribution appropriate for the function under consideration. As a result, the performance is poor for functions that are"anisotropic" in the sense that they have long and skinny level sets.

In an important step forward we have now devleoped the *Variable Metric Random Pursuit* (VRP) algorithm [19]. This algorithm does not sample from the unit sphere but from a nonuniform Gaussian distribution that is continuously adapted to fit an estimated Hessian. For quadratic functions (where the Hessian is constant), we prove concrete bounds for the number of iterations until the sampling distribution is such that it achieves nearly optimal progress rates.

These theoretical results are accompanied with careful experimental studies that assess the performance of VRP on a wide range of quadratic functions [17]. One particularly pleasing result is that the performance is less sensitive to the spectrum of the Hessian than it is for CMA-ES, a standard adaptive method that is very successful in practice [13].

We have also come closer to our goal of theoretically understanding derivative free methods for non-convex functions. The DIRECT algorithm is a well-known subdivision-based method to minimize Lipsschitz-continuous functions. We provide a first quantitative analysis of DIRECT that

gives bounds on the number of iterations before the minimum has been found up to a prescribed accuracy [4].

Task 1.b: Predicates and primitives

Many geometric algorithms reduce to a series of determinant computations. As the space dimension grows, a higher percentage of computation time is consumed by these predicates. We considered algorithms for convex hull, point location, and triangulation. We reveal and exploit structural similarity in the predicate matrices, in particular when matrices share many rows or columns in common. We developed two schemes to improve such predicates, implemented in CGAL. Both contributions are discussed in [7].

- In Period 1, we focused on sequences of determinants where the matrix entries that change belong to a specific row (or column), as in the case with pointsets lifted by different functions. The main idea was hashing of matrix minors so as to avoid duplicating partial results, and the main application has been the computation of indirectly-defined resultant polytopes in high dimension (up to 10), which is reduced to a sequence of regular triangulations in low dimension (up to 4).

  In this period, our optimized implementation led to a speedup factor of up to 100 in constructing resultant polytopes (Task 2.3a and [6]). The hashtable may be cleaned up when there is no available memory, with a time penalty. A more sophisticated technique (delete only a portion of the entries) can be performed if we use tries. However, experiments indicate this is not the most appropriate data structure, since it is much slower in practice. Our implementation was submitted to CGAL as a kernel wrapper named *Lifting_Kernel_d*; it enhances the current d-dimensional kernel. It was reviewed and the revised version is being prepared.

- More generally, we examined dynamic determinant algorithms for matrices that change by one row or one column; our first results appeared in [8]. We exploit existing formulae in linear algebra, expressing such a determinant in terms of known quantities and some computation. We establish quadratic or linear complexity for computing determinants encountered in convex hull or point location algorithms, respectively, instead of the record bound which is $O(k^{2.376})$, for $k \times k$ matrices. They outperform state-of-the-art determinant implementations, including CGAL, in most cases, offering a speedup factor of up to 3.5 and 78, in convex hull and point location methods, respectively, in dimension higher than 5. Our implementation is open-source (sourceforge). This result appeared in European Symposium of Algorithms (ESA 2012) [11].

Task 1.c: Adaptive data structures

Nearest neighbor search (NNS) is a fundamental problem, but is known to be hard. Hence the current emphasis on its approximation version: an approximation factor $\epsilon \geq 0$ is also given, and the answer is allowed to be at most ϵ times farther away than the true nearest neighbor. We study data structures and algorithms for three questions related to exploiting the explicit or implicit structure of the problem at hand.

- For dimensions 3-5, we focus on the Approximate Voronoi Diagram (AVD) [1], whose data structure is tree-based, and can optimize either query time or space usage. Moreover, it is meant to perform well with non-uniform data since space partition adapts to the data. Despite its theoretical performance, the AVD is known to be a "memory hog" even in small dimensions,

which daunted every attempt to implement it. We propose several enhancements in order to make it simpler and more space efficient, most notably: the discretization of spherical point clusters in general dimension, and the use of a quadtree instead of a bbd-tree to store the AVD. We offer a public-domain implementation in C++. Our preliminary experiments show our structure is faster and more accurate than state-of-the-art NNS libraries, such as ANN, CGAL, and Libnabo, albeit with high space usage (up to 100 times more than other methods) and time of construction. For example, when $d = 5, n = 2,000$ we built the AVD in 80hr; queries are $\geq 34\%$ faster than the other libraries.

- For high dimensions (e.g. 50, 100 or beyond) we develop a CGAL implementation of *randomized kd-trees* proposed in [16]. In particular, the coordinate at which the dataset is split at every tree level is chosen randomly among those for which the set shows greatest variance. A constant number of such trees are built, using different random choices; they are all searched for each query. These trees have been shown to exploit structure in datasets encountered in computer vision and image processing, typically in $\mathbb{R}^{128}$ [15]. We compare against dD Spatial searching in CGAL, which uses "lazy" kd-trees, and the randomized kd-trees in the FLANN library [15], which implement the same idea, albeit more generally and hence with larger overheads. Our preliminary experiments show that our data-structure is faster than CGAL, and comparable to FLANN. We expect to optimize our software and extend the experimental analysis. There are theoretical open questions related to randomized kd-trees, since no guarantee is known for their performance.

- A lot of research effort has been devoted to approximate NNS, but very little has considered how to exploit structure of the input dataset. We exploit nonrandom spatial patterns of data by studying almost aligned points, and show that queries take expected time in $O(\log\log(n))$, which is exponentially faster than if structure was ignored, while space usage is optimal. We assume the n points almost lie on $O(\log^t n)$ unknown lines, t and dimension d are fixed, the lines are distributed uniformly in a bounding sphere, and the points are uniformly distributed on every line. On the other hand, we can have any number of points per line at least as large as a polylog function in n. We further bound query time with high probability in $O(\log\log n\,(\epsilon^{-O(d^2)}\log\log n + f))$, where f is the cost of searching on a line. The crucial subproblem is to find the line nearest to a given point by transforming this problem to approximate NNS among points in dimension $O(d^2)$.

All contributions are discussed in [10].

Task 1.d: Machine learning techniques

We have used our *fastSVM* implementation in a data driven approach for hue preserving color blending. Color mapping and semitransparent layering play an important role in many visualization scenarios, such as information visualization and volume rendering. Color, more specifically hue, is used in many applications to encode nominal attributes. Standard techniques for semitransparent layering like the Porter-Duff over operator can induce new hues, i.e., false colors, that do not correspond to any value of the nominal attribute. Hence we have been looking for color blending operator that avoids false colors while retaining depth ordering and blending vividness. Our approach has been to learn such an operator from feedback that we gathered in a web based user survey. The survey provided us with data points in seven dimensional Euclidean space. The coordinates of the data points are the three color coordinates (in a perceptual uniform color space) of the two colors

to be blended plus an α-value for weighting the two colors. From the data points we have learned a function from $\mathbb{R}^7$ to $\mathbb{R}^3$ that maps a blending problem to a blended color. Learning this function boils down two solving one support vector machine problem and two support vector regression problems. The blending operator that we have obtained from our data driven approach turned out to give superior results in applications like parallel coordinate plots and volume rendering in the sense that it actually preserves hue and depth order perception while still being vivid, i.e., not a neutral gray.

Deviations: TUDO has not yet started to investigate machine learning techniques (Task 1.d). The main reason is that Christiane Lammersen who worked on the project in the first year has obtained a postdoc position at another university that started in September 2011 and the visa process for the new project member Ebrahim Ehsanfar took a bit longer than expected. We plan to address machine learning techniques in the period of the project.

Task 2.b: Compact representation of cell complexes

We have continued working on our implementation of an algorithm that computes the Hasse diagram underlying the flow complex of a point cloud. The algorithm has three phases: 1) compute one maximum of the distance function, 2) traverse the maxima graph, i.e., the graph that connects all maxima through index $(d-1)$-saddle points, and 3) explore all predecessors of a critical point (if this routine is called recursively then the whole flow complex is computed assuming we know all maxima). The maxima can either be finite or proxies for the maximum at infinity. We now use a better encoding of the maximum at infinity that makes our algorithm more efficient and more robust. Also, our initial implementation suffered from some numerical problems that we had mitigated by explicit perturbations of the input. Meanwhile we have essentially removed these numerical problems by using a different implementation of the primitives that are used by the algorithm. In the next period we want to focus on improving the parallelization of our implementation and also on applications of the Hasse diagram of the flow complex.

Simplicial complexes are widely used in combinatorial and computational topology, and have found many applications in topological data analysis and geometric inference. A variety of simplicial complexes have been defined, for example the Čech complex, the Rips complex and the witness complex. However, the size of these structures grows very rapidly with the dimension of the data set, and their use in real applications has been quite limited so far. As of now, there is no data structure for general simplicial complexes that scales to dimension and size. Best implementations have been restricted to flag complexes, a special type of simplicial complexes whose combinatorial structure can be deduced from their graphs.

Our approach aims at combining both generality and scalability. We propose a tree representation for simplicial complexes. The nodes of the tree are in bijection with the simplices (of all dimensions) of the simplicial complex. In this way, our data structure explicitly stores all the simplices of the complex but does not represent explicitly all the adjacency relations between the simplices, two simplices being adjacent if they share a common subface. Storing all the simplices provides generality, and the tree structure of our representation enables us to implement basic operations on simplicial complexes efficiently, in particular to retrieve incidence relations. Moreover, storing exactly one node per simplex ensures that the size of the structure adapts to the intrinsic complexity of the simplicial complex to be represented.

We provide a theoretical complexity analysis as well as detailed experimental results. We more specifically study Rips and witness complexes. Our code outperforms (sometimes by several orders of magnitude) the JPlex library (http://comptop.stanford.edu/u/programs/jplex/), a widely used

Java software package which can be used with Matlab. Our code can construct and represent both Rips and relaxed witness complexes of up to several hundred million faces in high dimensions, on various datasets. This work received the best paper award at the European Symposium on Algorithms ESA 2012 [3].

Task 2.c: Delaunay type complexes

The theory of optimal size meshes gives a method for analyzing the output size (number of simplices) of a Delaunay refinement mesh in terms of the integral of a sizing function over the input domain. The input points define a maximal such sizing function called the feature size. We present a way to bound the feature size integral in terms of an easy to compute property of a suitable ordering of the point set. The key idea is to consider the pacing of an ordered point set, a measure of the rate of change in the feature size as points are added one at a time. In previous work, Miller et al. showed that if an ordered point set has pacing ϕ, then the number of vertices in an optimal mesh will be $O(\phi^d n)$, where d is the input dimension. We give a new analysis of this integral showing that the output size is only $\Theta(n + n \log \phi)$. The new analysis tightens bounds from several previous results and provides matching lower bounds. Moreover, it precisely characterizes inputs that yield outputs of size $O(n)$.

Task 3.a: Silhouettes of polytopes

In Period 1, we designed an output-sensitive algorithm to compute the resultant polytope of pointset A or, more generally, its projection Π along a given direction. An important application is in reducing implicitization of parametric (hyper)surfaces to linear algebra, even in the presence of base points [9]. The resultant polytope can be defined as a Minkowski summand of the secondary polytope $\Sigma(A)$, but the latter is very much larger. Our algorithm computes the minimum number of secondary vertices, which correspond to the exact vertex set of the resultant silhouette; the latter projects bijectively to Π. Silhouette vertices are seen as representatives of equivalence classes on the secondary vertices.

In this period, we optimized this algorithm and its implementation. We prove the runtime is proportional to the number of output vertices and facets, thus establishing its output-sensitive nature. The algorithm is implemented in C++. Our core implementation is public-domain (sourceforge) and relies on the CGAL experimental package `triangulation`. The similarity between successive predicates is exploited by hashing computed minors (Task 2.1b, and [7]). We compute Π in $\mathbb{R}^6$ or $\mathbb{R}^7$ with $23,000$ or 500 vertices, respectively, within 2hr, which is faster than competitive software. Our results appeared in the ACM Symposium on Computational Geometry (SoCG 2012) [6].

One original feature of our approach is that we decided to represent Π by an optimization oracle, see Milestone MS10. In the related technical report [5] we emphasize the role of oracles, which allow us to exploit certain properties of the polar dual polytope and to extend our approach to (projections of) secondary and discriminant polytopes. In particular, our oracle-based approach should lead to a randomized polynomial-time volume approximation method, which is ongoing work in collaboration with B. Gärtner (ETHZ). The output of the current algorithm is a polytope in V- and H-representations along with its triangulation; the latter comes at no extra cost from the beneath-beyond algorithm for convex hull. We are currently investigating whether Π can be computed faster in some alternative representation by exploiting information on the resultant polytope's edges.

Deviations: Some underspending at NKUA to compensate for overspending in Year 1: 3 funded

person-months, 2 unfunded. This brings funded and unfunded PMonths, as well as work progress, at the levels of Annex I.

Task 3.b: Extreme Points

Building on the Master's thesis of Helbling [14], we have further investigated the structure of sparse point sets with respect to their extreme points. The major open question is whether the extreme points of a point set where each point has at most two nonzero coordinates can be computed faster than using general methods (the case of at most one nonzero coordinate is trivial [14]). In ongoing work, we could gather evidence that the case of two nonzero coordinates may be more complicated than expected: we could construct such a point set with the property that the intersection with a suitable two-dimensional plane has exponentially many vertices. Previously, such a construction was known only for points with at most three nonzero coordinates, while it is known that for points with at most one nonzero coordinate, such "nasty" behavior is impossible [12].

Task 3.c: Restricted Delaunay triangulations and variants

For a submanifold of Euclidean space, the restricted Delaunay complex, which is defined by the ambient metric restricted to the submanifold, was employed by Cheng et al. as the basis for a triangulation. However, it was found that sampling density alone was insufficient to ensure a triangulation, and manipulations of the complex were employed.

In an earlier work, Leibon and Letscher [SoCG 2000] announced sampling density conditions which would ensure that the Delaunay complex defined by the intrinsic metric of the manifold was a triangulation. In fact, the stated result is incorrect: sampling density alone is insufficient to guarantee an intrinsic Delaunay triangulation. Topological defects can arise when the vertices lie too close to a degenerate or "quasi-cospherical" configuration.

We have addressed this problem with the introduction of a parameterized notion of genericity for Delaunay complexes. Our interest in the intrinsic Delaunay complex stems from its close relationship with other Delaunay-like structures that have been proposed in the context of non-homogeneous metrics. For example, anisotropic Voronoi diagrams and anisotropic Delaunay triangulations emerge as natural structures when we want to mesh a domain while respecting a given metric tensor field.

Our paper [2] builds over our preliminary results on anisotropic Delaunay meshes and manifold reconstruction using the tangential Delaunay complex. The central idea in both cases is to define Euclidean Delaunay triangulations locally and to glue these local triangulations together by removing inconsistencies between them. We view the inconsistencies as arising from instability in the Delaunay triangulations, and provide explicit bounds on the stability with respect to the genericity parameter.

The stability of Delaunay triangulations has not previously been studied in this way. Parameterizing the genericity of a point set allows to quantify robustness in the Delaunay triangulation when either the points or the metric are perturbed. We then use these results to demonstrate conditions under which the intrinsic Delaunay complex and the restricted Delaunay complex coincide and are manifold. These results can be extended to the case of the tangential complex and of the witness complex.

The next issue is to design efficient algorithms to ensure that a point set is sufficiently generic. This can be done by various means : by refining the point set, by weighting the points and using weighted Delaunay triangulations, or by a simple perturbation scheme.

Milestones

MS 8: Decide whether approximate nearest neighbor data structures can speed up CMA-ES

As already foreseen at the end of Period 1 (se MS6), we now believe that a quantitative analysis of CMA-ES is out of the scope of this project. In Period 2, we therefore focused on the development of other adaptive methods and succeeded in analyzing one such method, the Variable Metric Random Pursuit [19]. In Period 3, we will fully concentrate on extending our results on Variable Metric Random Pursuit and related method; we will no longer systematically work on CMA-ES. In that sense, this milestone is obsolete.

MS 9: Decide on representations for tangential complexes

In low dimensions, the tangential complex can benefit from the very efficient implementation of Delaunay triangulations in CGAL. We have implemented an algorithm to construct 2-dimensional tangential complexes using the 2-dimensional Delaunay triangulations of CGAL. However, this representation is less general and flexible than the simplex tree described above. Both representation have their pros and cons, depending on the application. In future, we intend to provide both representations. However, currently, most of our efforts are directed towards more tractable simplicial complexes like the Rips, the witness and the relaxed Delaunay complexes.

MS 10: Decide on representations for indirectly defined polytopes

We examine indirectly-defined polytopes for which neither the (vertex) V-representation nor the (halfspace) H-representation is given; their vertices are specified as equivalence classes of a larger polytope. In computing such polytopes, we rejected the edge-skeleton representation because of its costly construction. We decided to focus on oracle-based representation and designed efficient algorithms for producing both V- and H-representations and the polytope's triangulation, in an output-sensitive way.

Bibliography

[1] S. Arya, T. Malamatos, and D.M. Mount. Space-time tradeoffs for approximate nearest neighbor searching. *J. ACM*, 57(1), 2009.

[2] Jean-Daniel Boissonnat, Ramsay Dyer, and Arijit Ghosh. Stability of Delaunay-type structures for manifolds. In *Proc. 28th ACM Symposium on Computational Geometry*, 2012.

[3] Jean-Daniel Boissonnat and Clément Maria. The Simplex Tree: An Efficient Data Structure for General Simplicial Complexes. In *Proc. 20th European Symposium on Algorithms*, volume 7074 of *LNCS*, 2012.

[4] Y. Brise and B. Gärtner. Convergence rate of the direct algorithm. Technical Report CGL-TR-47, ETH Zürich, 2012.

[5] I.Z. Emiris, V. Fisikopoulos, C. Konaxis, and L. Peñaranda. An oracle-based, output-sensitive algorithm for projections of resultant polytopes. Technical Report CGL-TR-28, NKUA, 2012.

[6] I.Z. Emiris, V. Fisikopoulos, C. Konaxis, and L. Peñaranda. An output-sensitive algorithm for computing projections of resultant polytopes. In *Proc. ACM Symp. on Computational Geometry*, pages 179–188, Chapel Hill, North Carolina, 2012. Full version submitted for journal publication.

[7] I.Z. Emiris, V. Fisikopoulos, and L. Peñaranda. High dimensional predicates: Algorithms and software. Technical Report CGL-TR-27, NKUA, 2012.

[8] I.Z. Emiris, V. Fisikopoulos, and L. Peñaranda. Optimizing the computation of sequences of determinantal predicates. In *Proc. European Workshop Comp. Geometry*, Assisi, Italy, 2012.

[9] I.Z. Emiris, T. Kalinka, C. Konaxis, and T. Luu Ba. Sparse implicitization by interpolation: Characterizing non-exactness and an application to computing discriminants. *J. CAD, Special Issue on Solid & Physical Modeling*, October 2012.

[10] I.Z. Emiris, A. Konstantinakis-Karmis, D. Nicolopoulos, and A. Thanos-Filis. Data structures for approximate nearest neighbor search. Technical Report CGL-TR-29, NKUA, 2012.

[11] V. Fisikopoulos and L. Peñaranda. Faster geometric algorithms via dynamic determinant computation. In *Proc. 20th Europ. Symp. Algorithms*, pages 443–454, 2012.

[12] B. Gärtner, C. Helbling, Y. Ota, and T. Takahashi. Klee-Minty Cubes are also Goldfarb cubes. Manuscript, 2012.

[13] Nikolaus Hansen and Andreas Ostermeier. Completely derandomized self-adaptation in evolution strategies. *Evolutionary Computation*, 9(2):159–195, 2001.

[14] C. Helbling. Extreme points in medium and high dimensions. Master's thesis, ETH Zurich, Zurich, Switzerland, 2003.

[15] Marius Muja and David G. Lowe. Fast approximate nearest neighbors with automatic algorithm configuration. In A. Ranchordas and H. Araújo, editors, *Proc. VISAPP: 4th Intern. Conf. Computer Vision Theory & Appl.*, volume 1, pages 331–340, February 2009.

[16] C. Silpa-Anan and R. Hartley. Optimised KD-trees for fast image descriptor matching. In *Proc. Comp. Vision Patt. Recognition*, 2008.

[17] S. U. Stich and C. L. Müller. On spectral invariance of randomized hessian and covariance matrix adaptation schemes. Technical Report CGL-TR-46, ETH Zürich, 2012.

[18] S. U. Stich, C. L. Müller, and B. Gärtner. Optimization of convex functions with Random Pursuit. Technical Report CGL-TR-09, ETH Zürich, 2011.

[19] S. U. Stich, C. L. Müller, and B. Gärtner. Variable Metric Random Pursuit. Technical Report CGL-TR-44, ETH Zürich, 2012.

[20] S. U. Stich, C. L. Müller, and B. Gärtner. Optimization of convex functions with Random Pursuit. *SIAM J. Opt.*, 2012, accepted.