NSF CAREER Proposal:
Realistically Efficient Geometric Algorithms

Project Summary

Computational geometry is the branch of theoretical computer science devoted to the design, analysis, and implementation of geometric algorithms and data structures. Geometric problems arise naturally in any computational field that simulates or interacts with the physical world—computer graphics, robotics, geographic information systems, computer-aided design, and solid modeling, to name a few—as well as in more abstract domains such as combinatorial geometry and algebraic topology. Aside from their obvious practical significance, geometric algorithms enjoy a rich and satisfying mathematical structure, and their development often requires tools from mathematical disciplines such as combinatorics, topology, and algebraic geometry, as well as traditional computational tools.

The proposal outlines a challenging career development plan focusing on research in a broad cross-section of computational geometry, building on and significantly broadening the PI's successful work in the field over the last several years. The proposal describes four broad classes of problems of specific interest to the PI: (1) kinetic data structures for collision detection, (2) geometric range searching data structures for both static and moving points, (3) Delaunay and other triangulations of well-spread point sets, and (4) intersection and convex decomposition of 'smooth' polyhedra. The PI will also work on other geometric problems as they arise.

As usual in theoretical computer science research, a primary goal of the proposed work is to develop of algorithms and data structures that are as efficient as possible in the worst case. However, since worst-case theoretical efficiency is often a poor predictor of practical performance, an equally important goal of the PI's research is the development of *realistically* efficient algorithms. This will be accomplished on two fronts. First, the PI will develop algorithms under realistic input models, thereby exploiting useful combinatorial and geometric properties of common input data for even greater efficiency in practice. Second, the PI will design and analyze algorithms under models of computation appropriate for each problem; non-standard models of particular interest are the external memory model and the kinetic data structure model.

A key component of the proposed plan is the integration of the PI's approach to algorithm development into the computer science curriculum at the University of Illinois, including the revision of existing algorithms courses to include new results and techniques, as well as continued development of new courses in computational geometry.

Altogether, the expected outcome of the project will be the development of important new algorithmic techniques and results, cross-fertilization between computional geometry and other areas of computer science, and curricular innovations. At the same time, the proposed plan will build a strong foundation for the PI's continued work in computer science research and education, and will provide the PI with numerous opportunities for cross-disciplinary collaboration.

NSF CAREER Proposal: Realistically Efficient Geometric Algorithms

1 Results from prior NSF support

The PI was supported by an NSF Mathematical Sciences Research Postdoctoral Fellowship (DMS-9627683) from September 1996 to August 1998 and from July 1999 to August 1999. During the period of this award, the PI's research followed two overlapping directions: data structures and applications for geometric range searching [4, 5, 10, 90, 98, 99] and algorithms for moving data [4, 11, 39, 90, 100]. Some of the following results are explained in more detail later in the proposal.

1.1 Geometric range searching

A typical geometric range searching problem is to preprocess a set of point, so that later the points inside an arbitrary query region can be counted or reported quickly. The PI and Agarwal wrote an extensive survey of the current best algorithms and data structures for geometric range searching, both in theory and in practice [10].

Lower Bounds. Extending his earlier Ph.D. thesis research [94], the PI proved the first nontrivial lower bounds for online hyperplane and halfspace *emptiness queries* [99]. The PI defined a general class of data structures called *partition graphs*, which models almost all current range searching data structures. The lower bounds, which take the form of space-time tradeoffs, are optimal at least up to polylogarithmic factors for the cases of polylogarithmic query time, $O(n^d/\operatorname{polylog} n)$ space, and two-dimensional queries.

External Memory Computation. The PI and coauthors have developed several new efficient range searching data structures in the now-standard *external memory model* of computation [22, 196]. This model assumes a two-level memory hierarchy. One *I/O* transfers a block of B words from external memory to internal memory or vice versa. The efficiency of a data structure is measured by the number of blocks used to store the data and the number of I/Os used to answer a query. See page 4 for more detailed definitions.

The PI, working with Arge, Agarwal, Franciosa, and Vitter, developed the first efficient external data structures for halfspace reporting queries [5]. They obtained a worst-case optimal external data structure for two-dimensional queries, which uses $O(n/B)$ blocks and answers queries in $O(\log_B n + k/B)$ I/Os, where k is the number of reported points, and a near-optimal structure for three-dimensional queries. They also developed an external version of partition trees [152], which answer d-dimensional queries in $O(n^{1-1/d+\varepsilon} + k/B)$ I/Os using $O(n/B)$ blocks, or in $O(n^{1-1/\lfloor d/2\rfloor+\varepsilon} + k/B)$ I/Os using $O((n/B)\log_B n)$ blocks.

Arge, Agarwal, and the PI also developed the first efficient data structures to store linearly-moving points in the plane, so that timed rectangle queries can be answered quickly [4]. One such structure, based on partition trees [5], uses $O(n/B)$ space and answers queries in $O((n/B)^{1/2+\varepsilon} + k/B)$ I/Os. A second structure is an external version of kinetic range trees [32, 41] that can answer chronologically-ordered queries in an optimal number of I/Os, but must be updated $O(n^2/B)$ times over the history of the motion. We also developed tradeoffs between these two approaches, extensions to higher-dimensional queries and more complex motions, and similar data structures for approximate nearest neighbor searching. These results will be described in more detail in Section 4.

These were the first two papers published by the PI at PODS [4, 5]; both papers were invited to the special issues of *Journal of Computer and System Sciences* devoted to the symposium.

Applications of Range Searching. The PI and Eppstein [90] developed an algorithm for constructing the *straight skeleton* of a polygon, a variant of the medial axis introduced by Aichholzer *et al.* [24]. For an n-gon with r reflex vertices, the algorithm runs in time $O(n^{1+\varepsilon} + n^{8/11+\varepsilon} r^{9/11+\varepsilon})$, improving the previous best upper bound of $O(nr \log n)$. The algorithm uses a technique of Eppstein for maintaining extrema of binary functions along with several dynamic range searching data structures. This paper was invited to a special issue of *Discrete & Computational Geometry*.

Consider the following variant of the *hidden surface removal* problem: Given n disjoint triangles in $\mathbb{R}^3$ and p arbitrary sample points ('pixels') in the xy-plane, determine the first triangle directly above each pixel. The PI defined an implicit representation of the output called the *sampled visibility map* that adapts to local changes in image complexity, and whose complexity is bounded both by the number of pixels and by the complexity of the analytic visibility map [98]. The PI also developed an algorithm to construct sampled visibility maps in $O(n^{1+\varepsilon} + n^{2/3+\varepsilon} k^{2/3} + p)$ time, where k is the output size. This is nearly optimal in the worst case and comparable to the best output-sensitive algorithms for ray casting [19] and analytic hidden surface removal [17, 47].

1.2 Kinetic data structures

The PI and his colleagues developed several new results in the *kinetic data structure* model introduced by Guibas *et al.* [40, 118]. A kinetic data structure (KDS) maintains some useful information about a set of continuously moving objects, along with several *certificates* that collectively prove the correctness of this information. Certificates can become invalid due to the motion of the objects, at which time the KDS must be updated. A kinetic data structure is *compact* if it uses little space, *efficient* if it undergoes few certificate failures, *responsive* if it handles each certificate failure quickly, and *local* if it quickly handles a change in an object's flight plan. See page 4 for more detailed definitions.

Binary Space Partitions. Agarwal, Guibas, and the PI [11] developed the first kinetic binary space partition trees (BSPs) for continuously moving, possibly intersecting, line segments in the plane, and for continuously moving but disjoint triangles in 3-space, extending earlier results for disjoint segments [14]. The kinetic structures are based on new randomized static BSPs with small size and optimal $O(\log n)$ depth. For planar segments, their BSP has size $O(n \log n + k)$, where k is the number of intersecting pairs of segments, and for disjoint triangles in space, their BSP has optimal depth and size $O(n \log^2 n + k')$, where k' is the number of edge intersections in the xy-projection of the triangles. The kinetic BSPs are compact, responsive, local, and efficient.

Collision Detection. The PI and co-authors developed several new kinetic data structures for collision detection between two moving convex objects in two or three dimensions, or between two nonconvex objects in the plane [39, 100]. The performance of these structures is sensitive to the separation of the objects during their motion. Specifically, for convex objects, the performance of the data structures improved smoothly as the distance between them increases, and for nonconvex objects, the performance is a function of the number of links in a polygonal chain separating the objects. These results will be described in more detail in Section 3.

2 Overview

Computational geometry is the branch of theoretical computer science devoted to the design, analysis, and implementation of geometric algorithms and data structures. Geometric problems arise naturally in any computational field that simulates or interacts with the physical world—computer graphics, robotics, geographic information systems, computer aided-design, and molecular modeling, to name a few—as well as in more abstract domains such as combinatorial geometry and algebraic topology. Aside from their obvious practical significance, geometric algorithms and data structures enjoy a rich and satisfying mathematical structure, and their development often requires tools from mathematical disciplines such as combinatorics, topology, and algebraic geometry, as well as traditional computational tools.

The proposal outlines a challenging career development plan focusing on research in a broad cross-section of computational geometry, building on and significantly broadening the PI's successful work in the field over the last several years. Specific problem areas in which the PI plans to work include kinetic data structures, geometric range searching, spatial and temporal databases, external memory computation, mesh generation, ray shooting and hidden surface removal, surface reconstruction and approximation, collision detection and motion planning, and polyhedral decomposition. This classification is at best a rough guide, as many interesting geometric problems fall into more than one category.

As usual in theoretical computer science research, a primary goal of the PI's research is the development of algorithms and data structures that are as efficient as possible in the worst case. However, it has long been recognized that standard worst-case analysis often fails to predict the practical behavior of algorithms. For many geometric problems, the asymptotically fastest algorithm is hopelessly outmatched in practice by a simple heuristic with much worse theoretical performance. Several different factors contribute to this effect, including large constants hidden in $O(\cdot)$ notation, numerical instability, inability to deal with degenerate cases, reliance on inappropriate models of computation, and effects of contrived worst-case inputs. Although there is increasing attention to these issues in the computational geometry community, a large gap still remains between theory and practice. The PI believes that application areas can benefit tremendously from theoretical work, and likewise practical considerations are a strong motivator for theoretical investigation. Thus, an equally important goal of the PI's research is the development of *realistically* efficient algorithms.

2.1 Realistic input models

One of the most significant barriers to realistic efficiency is that algorithms are traditionally designed and analyzed for efficiency only in the worst case. For many problems, the worst-case inputs are uncommon or highly contrived. A key component of the proposed research is developing algorithms within *realistic input models*. The overall approach is to identify useful combinatorial and geometric features of realistic inputs, and develop algorithms to exploit those features.[1]

Exactly which features are both useful and realistic obviously depends on the problem at hand. For some of the problems discussed in this proposal, the PI has identified features of real-world input that allow algorithms and heuristics to perform significantly better in practice than current theory predicts. Some of the proposed models and conjectured upper bounds are suggested by worst-case examples and lower bound constructions. The PI's successful research in geometric lower bounds [101, 92, 97, 94, 95, 99] will be a tremendous advantage in this effort.

For problems involving closed shapes such as polygons and polyhedra, the most common realistic model assumes that all input objects are *fat*—intuitively, fat objects have no long skinny pieces [12, 15, 45, 87, 144, 136, 154, 164, 166, 192, 191, 206, 207]. De Berg *et al.* [48] and Vleugels [197] consider several

[1] These features are sometimes collectively called the *geometric complexity* of the input, in order to distinguish them from the ubiquitous "n", the combinatorial complexity of the input. Unfortunately, this term sometimes also refers to the complexity of geometric algorithms [61, 66, 147, 184] and in computer graphics [26] it usually means combinatorial complexity!

different input models similar to fatness, prove relationships between them, and present algorithms to compute their relevant parameters; see also [137, 190]. Several impressive results have been developed in this model, but for many problems, assuming fatness makes no appreciable difference in complexity. Two such problems are decomposing nonconvex polyhedra into convex pieces and detecting intersections between nonconvex polyhedra. To handle these problems, the PI proposes a new input parameter called *smoothness* as an alternative to fatness—a polyhedron with triangular faces is smooth if it has no sharp angles, either between edges or between facets. See Section 6 for details.

Fatness and its relatives are also inappropriate models for problems where the input is just a set of points. One such problem where practical behavior is significantly better than predicted by current theory is the construction of Delaunay triangulations. In an effort to explain this discrepancy, the PI considers the complexity of Delaunay triangulations as a function of the *spread* of a set of points, the ratio between the longest and shortest pairwise distances. Preliminary results are reported in Section 5.

Mitchell *et al.* [158] describe a *query-sensitive* ray shooting data structure, where the query time depends on the size of a minimal set of disks that cover the ray and are disjoint from the scene. (See [102, 159] for other versions of 'query sensitivity'.) The PI and coauthors have developed similar 'path-sensitive' results for detecting collisions between two convex objects [100]. Many other input parameters have also proven useful for collision detection, including the separation or entanglement of the objects, and the algebraic degree of their motion [100, 39]. These will be discussed in more detail in Section 3.

Many authors, including the PI, have developed online searching algorithms whose performance depends on the *coherence* or *similarity* of successive queries [4, 102, 127]. The PI's efforts in this direction are outlined in Section 4. That section of the proposal also introduces a new model of *optimal input-sensitive* range searching, where the goal is to be as efficient as possible for *every* input, not just in the worst case or in other special cases.

2.2 Appropriate computational models

Another barrier to realistic efficiency is over-reliance on standard models or modes of computation. Just as with input models, the appropriate computational model varies from problem to problem. Two non-standard models are of particular importance in the proposed work: the external memory model of Vitter and Aggarwal [22, 196] and the kinetic data structure model of Basch, Guibas, and Hershberger [40, 118]. Theoretical research in both of these models by the PI and others has led directly to improvements in practical performance of algorithms.

External memory. Applications are increasingly required to deal with massive amounts of geometric and other data. These large datasets are usually too large to fit in main memory and must be stored on some external device such as a disk. Since a single disk access can take as much time as millions of accesses to internal memory, communication between internal and external memory is the bottleneck in such computations, rather than actual cpu time. To address this issue, Aggarwal and Vitter [22] developed the now-standard *external memory* model of computation. In this model, data is stored on disks in *blocks* of B items each, and a single input/output operation, or *I/O*, transfers one block from external memory to internal memory or vice versa. The size of a data structure is measured in terms of the number of blocks used, and the running time of an algorithm is measured in number of I/Os. This model is now ubiquitous in the spatial database community and is gaining wide exposure in computational geometry as well [113], in part because of the PI's recent work [5, 4]. Theoretical and practical results in this model up to 1999 are surveyed by Vitter [196].

Kinetic data structures. Application areas that model the physical world often call for simulating a collection of continuously moving objects and maintaining some useful information as the simulation evolves. The traditional solution to this problem is to discretize time into short intervals and compute the desired information at every time step. The most effective algorithms are designed to exploit *temporal coherence*, maintaining information from one time step to the next instead of recomputing it from scratch, in the hope that only minor changes occur between steps. Choosing an appropriate time step is quite difficult. If time steps are too small, large amounts of computation are wasted determining that nothing has happened since the last step; if time steps are too large, the simulation may miss crucial events. Formal analysis of these discrete-time methods is virtually impossible.

A promising alternative to this approach is the *kinetic data structure* framework introduced in a seminal paper of Basch, Guibas, and Hershberger [40]. In the kinetic framework, each moving object follows a posted flight plan. A kinetic data structure (KDS) maintains a set of *certificates* that collectively prove the correctness of the information begin maintained. At various times, called *events*, certificates can fail, in which case the kinetic data structure must be updated. The failure time of a certificate can be computed from the flight plans of the objects. Instead of a global time step, the simulation is driven by an *event queue*, which stores the certificates sorted by their failure times. The kinetic model provides a framework for formal analysis of algorithms and data structures for moving objects. A good KDS is *compact* if it uses few certificates, *responsive* if it can be updated quickly after a certificate failure, *efficient* if the number of (unnecessary) events is small, and *local* if it adjusts easily to changes in the motion plans of the objects. Results in this model up to 1998 are surveyed by Guibas [118]; for more recent developments by the PI and others, see [4, 8, 7, 11, 13, 39, 100, 119, 125, 121, 130, 140].

2.3 Summary

The overall goal of the proposed research is to develop algorithms that are both theoretically and practically efficient. This research would have a significant impact both on the theoretical state of the art in computational geometry and on the practical use of geometric algorithms, by developing and exploiting the rich interplay between theory and practice. The increasingly practical direction of the PI's research, evidenced by recent well-received publications in major database conferences [5, 4], offers hope of direct impact on real-world applications. The proposed career development plan will also build a strong foundation for the PI's continued work in computer science research and education, and will provide the PI with numerous opportunities for cross-disciplinary collaboration.

The remainder of the proposal outlines four broad classes of open problems of particular interest to the PI, some with unpublished partial results. Section 3 considers kinetic data structures for collision detection that favorably exploit object separation and motion complexity. Section 4, describes several open problems in geometric range searching, including a new model for optimal input-sensitivity, dependence on temporal coherence of timed queries, and kinetization and externalization of existing data structures. Section 5 considers the complexity of three-dimensional Delaunay triangulations as a function of the *spread* of the input. Section 6 presents the PI's model of *smooth* polyhedra and poses a number of open problems which are difficult in other input models such as fatness. In addition to these problems, the PI will also work on other geometric problems as they arise in the course of his research and collaboration. Finally, Section 7 discusses the PI's continuing and proposed educational activities.

3 Collision detection

Collision detection is an algorithmic problem that arises naturally in every area of computer science dealing with the simulation of moving physical objects. Examples include motion planning in robotics, virtual reality animations, computer-aided design and manufacturing, and computer games. Efficient collision detection algorithms usually break the problem into two phases: a *broad phase*, in which 'close' pairs of objects are quickly selected for further consideration, and a *narrow phase*, in which each 'close' pair of objects is tested individually [132]. (A similar split is described in the spatial database literature, where the phases are called *filtering* and *refinement*, respectively [163].) The PI's collision detection research focuses on the narrow phase, which is specialized according to the types of objects being considered.

Kinetic data structures are a natural model for designing and analyzing collision detection algorithms. Although most collision detection techniques were not developed in the kinetic model, almost all of them can be translated easily into kinetic data structures for purposes of theoretical analysis and comparison. This translation will be made implicitly throughout the proposal.

Convex polytopes. The simplest objects to consider for collision detection are convex polytopes—polygons in the plane or polyhedra in 3-space [55, 77, 108, 149, 150, 155, 156, 157]. Effective techniques for this case maintain a separating hyperplane (line in 2d, or plane in 3d) between the objects; if no separating hyperplane exists, a collision has occurred. The most theoretically efficient techniques for convex collision detection were developed by the PI and his coauthors [100]. Unlike earlier techniques, these structures are provably sensitive to both the distance between the objects and the complexity of their motion as follows. Let $\mu = \min\{n, \sqrt{D/s}\}$, where n is the combinatorial complexity of the polygons, D is their maximum diameter, and s is their minimum separation. For convex polygons in the plane, for example, their technique maintains a single separating line that changes $O(\log \mu)$ times when the relative motion of the polygons is translation along a straight line or convex curve, $O(\mu)$ for translation along an algebraic trajectory, and $O(\mu^2)$ for algebraic rigid motion. Similar results are obtained in three dimensions; see also [120]. Even in the worst case, when objects are almost touching, these bounds are better than any earlier technique.

These KDSs are based on a hierarchy of approximations to the objects, similar to a Dobkin-Kirkpatrick hierarchy [82, 81] but with additional guarantees on approximation error. Specifically, each level is defined by an algorithm of Dudley [84] that approximates any d-dimensional convex body with relative error ε using a polytope with $O(1/\varepsilon^{(d-1)/2})$ facets. The PI is currently investigating whether different bounding shapes, such as intersections of balls, can be used to obtain tighter approximations, and thus more efficient kinetic data structures. Using spherical hierarchies instead of polytopes would make detecting collisions between rotating objects easier.

The event bounds for most kinetic structures have some hidden dependence on the algebraic degree of the motion, but intuitively, complex motions should only be expensive when the objects are close together. This intuition motivated the PI to develop structures that exhibit *hysteresis*—where, after a certificate failure has occurred, no other certificate failure can happen until the objects have moved by some constant fraction of their current separation. Using these structures, one can bound the number of events by the size of a certain cover of the motion path by balls, somewhat reminiscent of [158]. Developing hysteretic kinetic data structures for other problems is an interesting and challenging open problem, which the PI plans to pursue.

Simple polygons. The PI and coauthors also designed the first kinetic data structure for detecting collisions between two moving simple polygons [39]. This kinetic data structure maintains a planar subdivision of the free space between the two polygons called the *external relative geodesic triangulation* (ERGT). Like the convex polygon structure described above, this KDS is *separation sensitive*: The number of certificates maintained at any time is $O(\kappa \log n)$, where κ is the length of the minimum-link separator between the polygons. The KDS is also responsive and efficient. Later papers generalize this structure to multiple polygons [13, 140].

The PI discovered the following simple method to combine the techniques from [100] and [39] to obtain a KDS sensitive to both distance and separator complexity. One can add n new vertices to any polygon so that the Voronoi region of every edge (restricted to the convex hull) has area at most D^2/n, where D is the polygon's diameter. In a kinetically maintained ERGT of such augmented polygons, the number of active certificates drops from $O(\kappa \log n)$ to $O(\kappa \log \mu)$, where $\mu = \min\{n, \sqrt{D/s}\}$ as above. The PI conjectures that the number of events is also a function of the minimum distance between the objects, and that similar modification will add separation-sensitivity to the more recent kinetic structures [13, 140].

Nonconvex polyhedra. Most collision detection applications involve three-dimensional non-convex polyhedra, but little is known theoretically about this case. Existing techniques for simple polygons do not generalize to three dimensions. In particular, the most natural generalization of the ERGT would have non-planar facets that warp during motion. The most popular practical approach for this case is to approximate each object by a *hierarchy of bounding volumes*. For a given placement of two disjoint objects, the algorithms refine the objects' hierarchies only to the coarsest level at which the resulting bounding volumes are disjoint. Several types of bounding volume hierarchies have been proposed and implemented [38, 157, 114, 133, 141, 204], all of which can be recast and analyzed in the kinetic data structure framework. Unfortunately, it is possible for two polyhedra to be positioned so that almost every edge of one is arbitrarily close to almost every edge of the other. In this case, any existing volume hierarchy technique will require $\Omega(n^2)$ certificates, one for each close pair of edges. Intuitively, the performance of bounding volume hierarchies depends on the number of 'close' pairs of edges between the two objects; formalizing an proving such a relationship for any hierarchy is an interesting open problem. (This issue will be addressed again in Section 6.) The PI also plans to investigate 'self-adjusting' hierarchies that change on the fly as the objects move, either by adjusting the shape of the individual bounding volumes, perhaps using the PI's convex separator techniques [100], or by modifying the combinatorial structure of the hierarchy itself, as was recently done with binary space partition trees [35, 36].

Motion planning. The PI is currently working with Jean Ponce at UIUC and Jean-Daniel Boissonnat, Jean-Pierre Merlet, and Mariette Yvinec at INRIA on applying kinetic collision-detection data structures to the important domain of motion planning of mobile and articulated robots. This work is being funded under a joint agreement between UIUC, CNRS, and INRIA.

4 Geometric range searching

A typical *geometric range searching* problem has the following form: Preprocess a set of points into a small data structure, so that the points lying inside a query region can be quickly counted or reported. Common types of query regions are axis-parallel boxes, halfspaces, simplices, and spheres. The recent survey by the PI and Agarwal described most of the known results in this area [10]. Although most of the basic questions about range searching have been answered, there are still several interesting open problems.

Optimal input-sensitivity. One of the simplest geometric range searching problems was first posed by John Hopcroft about 20 years ago: Given a set of n points and n lines in the plane, does any point lie on a line? In terms of worst-case complexity, Hopcroft's problem is completely solved—the fastest algorithm known runs in time $n^{4/3}2^{O(\log^* n)}$ [153], and the PI proved a nearly matching $\Omega(n^{4/3})$ lower bound [95]. However, there is room for further improvement. There are many classes of inputs for which Hopcroft's problem is easier to solve; for example, there are easy $O(n \log n)$-time algorithms if the points are in convex position, or if the lines all are tangent to a convex curve. A natural open question is the existence of *optimal input-sensitive* algorithm, that is, one whose running time is optimal for *every* input, not merely for the worst-case input.

The PI believes that the correct measure of optimality is suggested in his $\Omega(n^{4/3})$ lower bound proof [95]. Let P be a set of points and L a set of lines, with no point on any line. A *monochromatic minor* of P and L is a pair of subsets $P' \subseteq P$ and $L' \subseteq L$ such that P' lies either above every line in L' or below every line in L'. The *size* of the minor is $|P'| + |L'|$. A *monochromatic cover* is a collection of monochromatic minors such that every point-line pair belongs to at least one minor. Finally, the *cover complexity* $\mu(P, L)$ is the minimum total size of any monochromatic cover. The PI proved that $\mu(P, L) = \Omega(n^{4/3})$ in the worst case and, more importantly for this proposal, that any algorithm for Hopcroft's problem requires $\Omega(\mu(P, L))$ time given P and L as input. The question of optimal input-sensitivity can thus be formalized as follows: Is there an algorithm that runs in time $O(\mu(P, L))$, for all input sets P and L?

A similar but slightly weaker notion of input-complexity can be defined for online query problems. Consider the case of *line queries*: given a line ℓ, does it contain a point in P? The PI proved an optimal lower bound $st = \Omega(n^2)$ for data structures of size s that answer line queries in time t [99]. Let $\mu(P, k) = \max_{|L|=k} \mu(P, L)$, where the maximum is taken over all sets L of k lines. The PI's lower bound proof implies that for any integer k, either $s = \Omega(\mu(P, k))$ or $t = \Omega(\mu(P, k)/k)$. Is there a family of online data structures that match these bounds for any k?

Some recent progress in this direction is provided by Har-Peled and Sharir [127], who describe an algorithm that can solve Hopcroft's problem in time $O((n + w)\alpha(n) \log n)$, where w is the minimum number of crossings between the lines and a spanning tree of the points. While this algorithm is input-sensitive, it is not optimal, since $w = \Theta(n^{3/2})$ in the worst case [72]. Cover complexity also appears to be related to the *hereditary discrepancy* of a set of points and lines, which Chazelle and Lvov [69] have used to prove an $\Omega(n \log n)$ lower bound on the arithmetic circuit complexity of a variant of Hopcroft's problem. See [62, 64, 65, 71] for related results. The PI plans to investigate these connections more closely.

Hopcroft's problem and line queries are the simplest special cases of more general problems such as higher-dimensional halfspace and simplex range searching [153], point location [16, 124], constructing cells in arrangements [1, 19], ray shooting [2, 17, 19, 44, 93], hidden surface removal [19, 43, 46, 44, 47, 98, 111, 138, 160, 171, 185], and collision detection [68, 93, 170, 181]. (Some of these connections are explored further later in the proposal.) Efficient solutions for these problems were developed using tools originally applied to Hopcroft's problem and line queries, so it is natural to expect better input-sensitive algorithms for these special cases to lead to better algorithms for these more general problems as well.

External memory. Geometric range searching is one of the most common operations in geographic information systems and other spatial databases. A tremendous amount of work has been done in the database community on data structures for geometric range searching, which they usually call *spatial data structures* [179, 180]. Recent results in this area are surveyed by Nievergelt and Widmayer [161]. Because modern spatial databases store massive amounts of data, in some cases approaching hundreds of terabytes, these data structures must be designed and analyzed in the external memory model defined on page 4, and only data structures using the minimum $O(n/B)$ blocks to store n objects are considered truly practical.

While spatial data structures with good practical performance have been known for many years, most classical structures have terrible worst-case behavior. On the other hand, many worst-case efficient data structures have been developed in the computational geometry community that do not perform well in practice, in part because they assume an internal memory model. Worst-case efficient external structures have only been developed for a few simple types of queries: two and three-dimensional orthogonal queries [32, 116, 115, 195], planar point location [3, 33, 113], and halfspace queries in two and three dimensions.

The first efficient external structures for this last problem were developed by the PI and coauthors [5]. These included an optimal structure for two-dimensional queries, which answers queries in $O(\log_B n + k/B)$ I/Os, where k is the number of reported points, and a near-optimal structure for three-dimensional queries. Through the standard paraboloid lifting map $(x, y) \mapsto (x, y, x^2 + y^2)$, these are also the first near-optimal external structures for circle queries and k-nearest neighbor queries in the plane. This paper also introduced an external version of partition trees [152], which can answer d-dimensional queries in $O(n^{1-1/d+\varepsilon}+k/B)$ I/Os using $O(n/B)$ blocks, and in a subsequent paper, the PI and coauthors introduced *multilevel* external partition trees for finding the set of lines intersecting a query rectangle in $\mathbb{R}^3$ [4].

Several online search problems still have no worst-case efficient external solutions, including orthogonal range searching in dimensions higher than three, window queries, and ray-shooting queries. The PI is currently working on some of these problems with Pankaj Agarwal and Lars Arge at Duke.

Kinetic range searching. Maintaining databases of moving objects has recently attracted a lot of attention in the database community [76, 188, 189, 177, 174, 178, 188, 193, 202, 198, 200, 201, 199], but with a few exceptions [4, 143, 174, 177] it has focused mostly on developing data models and query languages rather than efficient searching algorithms.

Agarwal, Arge, and the PI developed the first efficient data structures to store linearly-moving points in the plane, supporting queries of the form "Which points were (or will be) in rectangle R at time t?" [4]. Two basic data structures were developed. One structure, based on external partition trees [5], answers queries in $O((n/B)^{1/2+\varepsilon} + k/B)$ I/Os for points in any fixed dimension. The other is an external version of kinetic range trees [32, 41] that can answer queries in $O(\log_B n + k/B)$ I/Os, even for more complex motions, but must be updated $O(n^2/B)$ times over the history of the motion and can only handle queries in chronological order. Combining these two approaches yields interesting tradeoffs between query time and kinetic update frequency, analogous to standard space-time tradeoffs [10, 153] but always using near-linear space.

Even in the internal memory setting, very little is known about non-orthogonal kinetic range searching. The PI plans to investigate kinetic halfspace, simplex, and nearest neighbor queries in both the internal and external memory models. Kinetic nearest neighbor queries are especially interesting, in part because of their real-life applicability to problems in traffic management and mobile networking [142], and in part because simple near-solutions already exist. Agarwal, Arge, and the PI have developed external kinetic structures for *approximate* nearest neighbor queries, whose cost depends on the desired level of approximation. The PI has also proved that the history graph of a randomized incremental algorithm for building Voronoi diagrams [122, 86] can be easily kinetized. The resulting KDS uses $O(n \log n)$ space, handles events in $O(\log n)$ time, and answers queries in $O(\log^2 n)$ time; the number of kinetic events is only a constant times the number of combinatorial changes to the Voronoi diagram [25, 123]. Unfortunately, this KDS is not local—changing the motion of a point with a complex Voronoi cell could require $\Omega(n \log n)$ time. An *implicit* representation of the Voronoi diagram seems to required here, perhaps similar to recent dynamic planar convex hull data structures [54, 58, 116, 165]. The PI believes that ideas developed by Agarwal and Matoušek [18] for similar *dynamic* problems will be useful here.

Kinetic persistence and query coherence. The kinetic range tree can only answer orthogonal range queries for points in their current positions, but by adding *persistence* [83], one can allow questions about past or future positions of the points as well. The PI, Arge, and Agarwal developed a *temporarily persistent* version of the external kinetic range tree that can answer questions about the past or future, as long as $O(n/\log_B n)$ events have occurred between the query's time stamp t_q and the current time (*now*), with only a constant increase in size, query time, and kinetic update time. They then constructed a multilevel data structure by adding persistent kinetic range trees as secondary structures to a partition tree. The resulting structure answer queries arbitrarily in the past or future, at a cost that depends monotonically on $|now - t_q|$. Thus, the time required to process a sequence of queries depends on its *coherence*—how closely the time stamps t_q are to chronological order. If the points and their trajectories are uniformly distributed, the expected time for a single query is $O((\Delta/n)^{1/2+\varepsilon} + k/B)$ I/Os, where Δ is the number of kinetic events between t_q and *now*. The PI conjectures that this query time can be achieved in the worst case using a slightly different data structure. The PI also plans to apply this technique to other kinetic query structures.

5 Triangulations of well-spread points

Delaunay triangulations are one of the most basic and commonly used structures in computational geometry. They have a number of useful properties that make them ideal for use in finite-element mesh generation, geographic information systems, interpolation, and surface reconstruction [162, 106]. In fact, the practical success of Delaunay triangulations seems to far exceed current theoretical predictions. The Delaunay triangulation of n points in $\mathbb{R}^3$ can have $\Theta(n^2)$ features in the worst case, but this quadratic behavior is never observed in practice; Delaunay triangulations of real-world data almost always have linear complexity. Dwyer [85] has shown that the Delaunay triangulation of uniformly distributed points in a ball has only linear complexity, and Golin and Na recently derived similar results for points uniformly distributed on the surface of certain convex polytopes in $\mathbb{R}^3$ [110]. Although these results are encouraging, their practical relevance is highly questionable—real world data is *not* random. In an effort to better explain the discrepancy between theory and practice, the PI is investigating the complexity of three-dimensional Delaunay triangulations under geometric assumptions about the input point set.

The *spread* of a set of points (also called the *distance ratio* [78]) is the ratio between the largest and smallest pairwise distances. If the spread takes its minimum value $\Theta(n^{1/3})$, the points are packed into a tight lattice, and the Delaunay triangulation has only linear complexity. On the other hand, all known examples of point sets with quadratic-complexity Delaunay triangulations have spread $\Omega(n)$. Thus, it is natural to ask how the complexity of the Delaunay triangulation changes as the spread varies between these two extremes. The spread of a set of points is loosely related to its 'dimensionality'. If a set of points uniformly cover a bounded region of space, a surface of bounded curvature, or a curve of bounded curvature, its spread is respectively $\Theta(n^{1/3})$, $\Theta(n^{1/2})$, and $\Theta(n)$. The case of surface data is particularly interesting in light of numerous algorithms that reconstruct surfaces using a subcomplex of the Delaunay triangulation [27, 28, 29, 30, 23, 51, 52, 109, 126].

The PI is currently investigating this question, and he believes that the correct complexity bound is given by the following conjecture.

Conjecture 5.1. *The Delaunay triangulation of n points in $\mathbb{R}^3$ with spread Δ has complexity $O(\Delta^{3/2}n^{1/2})$.*

Note that if this conjecture is true, then the commonly used randomized incremental flipping algorithm constructs 3d Delaunay triangulations in only $O(\Delta^{3/2}n^{1/2}\log n)$ time [79, 86]; see also [59]. The following construction of the PI shows that this is the best possible upper bound. Consider a set P consisting of $k = (n/\Delta)^{3/2}$ copies of a standard 'building block' B, placed on a regular cubical lattice with constant distance between adjacent blocks. The points in B lie on two skew line segments of constant

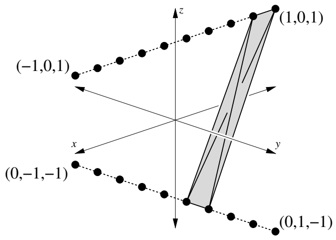

Figure 1. The 'building block' B and one of its Delaunay tetrahedra.

length and separation, with $n/2k$ evenly-spaced points on each segment. The Delaunay triangulation of B has complexity $\Omega(n^2/k^2)$. See Figure 1. The diameter of P is $O(k^{1/3})$, and its closest pair of points is at distance $\Omega(k/n)$, so its spread is $O(n/k^{2/3}) = O(\Delta)$. The spacing between blocks can be chosen so that the Delaunay triangulation of each block is a subcomplex of the Delaunay triangulation of P, so altogether P has $\Omega(nr) = \Omega(\Delta^{3/2}n^{1/2})$ Delaunay tetrahedra.

The PI is attempting to prove this conjecture by classifying the edges of the Delaunay triangulation into two classes and counting each class separately. (Euler's formula implies that counting edges is sufficient.) Specifically, let P be a set of n points with closest distance 1 and diameter Δ, and let r be some real number between 1 and Δ. Say that an edge in the Delaunay triangulation of P is *short* if its length is at most r, and *long* if its length is at least r.

Conjecture 5.2. *The Delaunay triangulation of* P *has* $O(nr)$ *short edges and* $O(\Delta^3/r)$ *long edges.*

Curiously, the number of long edges does not seem to depend on the number of points! It would suffice to prove these upper bounds only for the specific value $r = O(\Delta^{3/2}/n^{1/2})$, but the PI believes they hold for all values of r. If so, they are tight. Two simple special cases are especially interesting. Setting $r = 1$, so every edge is long, gives us the conjectured upper bound $O(\Delta^3)$; setting $r = \Delta$, so every edge is short, gives us the conjectured upper bound $O(n\Delta)$. Proving either of these two special cases will give significant insight into the general case. Instead of classifying edges, one could also classify triangles or tetrahedra based on their circumradius, diameter, or longest edge length. The PI believes that no matter how such objects are classified, there are $O(nr)$ 'short' objects and $O(\Delta^3/r)$ 'long' objects; these are the best possible bounds.

The PI has already proven a weaker upper bound using a similar approach. The nontrivial proof of the following lemma is omitted from this proposal.

Lemma 5.1. *Any point in* P *is an endpoint of* $O(r^2)$ *short edges, and* $O(\Delta^3/r)$ *points in* P *are endpoints of long edges.*

Theorem 5.2. *The Delaunay triangulation of* P *has complexity* $O(\Delta^4)$.

Proof: Let $L(r) = O(\Delta^3/r)$ be the number of endpoints of long edges. If the farthest neighbor of a point p is at distance between r and $r+1$, then p has $O(r^2)$ neighbors. Thus, the number of edges is $\sum_{r=1}^{\Delta} O(r^2)\cdot(L(r)-L(r+1)) = \sum_{r=1}^{\Delta} O(r)\cdot L(r) = \sum_{r=0}^{\Delta} O(\Delta^3) = O(\Delta^4)$. □

As mentioned earlier, a motivating application of this line of research is surface reconstruction. Let Σ be a closed manifold without boundary embedded in $\mathbb{R}^3$. The *medial axis* of Σ is the set of points with more than one closest point on Σ. The *local feature size* of a point $x \in \Sigma$, denoted $\mathrm{lfs}(x)$, is its distance to the medial axis. Finally, an ε*-sample* of Σ is a set of points such that for any point $x \in \Sigma$, the distance from x to the nearest sample point is at most $\varepsilon \cdot \mathrm{lfs}(x)$. Amenta and others have shown that for all ε less than some absolute constant, a surface homeomorphic to Σ can be reconstructed from any ε-sample [30]. In this context, the PI believes that Conjecture 5.1 can be further strengthened as follows.

Conjecture 5.3. *The Delaunay triangulation of an ε-sample of a connected surface has complexity* $O(n + \Delta^2)$.

This conjecture is consistent with the recent probabilistic results of Golin and Na [110]. Another simple construction of the PI establishes the matching lower bound. The restriction to connected surfaces is necessary—a modification of the earlier lattice construction gives an ε-sample of a surface with $(n/\Delta)^{3/2}$ components. The dependence on spread is also necessary, since for any $\varepsilon > 0$ and for any surface except a sphere, there are ε-samples whose Delaunay triangulations have size $\Omega(n^2)$.

Finally, a more surprising conjecture of the PI concerns *arbitrary* triangulations.

Conjecture 5.4. *Any triangulation of n points in $\mathbb{R}^3$ with spread Δ has complexity* $O(n\Delta)$.

Once again, as the following construction of the PI shows, this is the best upper bound possible. Start with n/Δ vertical line segments of length Δ, whose lower endpoints lie on a square integer lattice in the xy-plane. Perturb the line segments slightly, so that no two segments are parallel. On each line segment, place Δ evenly spaced points. Between every nearby pair of segments, construct a product-like triangulation of complexity $\Omega(\Delta^2)$, similar to the 'building block' in Figure 1. Finally, triangulate the remaining tunnels. The resulting triangulation has complexity $\Omega(n\Delta)$.

Of course, many of these questions can be generalized easily to higher dimensions; the PI will consider these questions in full generality once the three-dimensional case is settled satisfactorily.

6 Smooth polyhedra

Nonconvex three-dimensional polyhedra are ubiquitous in computer graphics, solid modeling, computer aided design and manufacturing, robotics, and other geometric application areas. Unlike nonconvex polygons in the plane, for which most problems can be solved easily and efficiently, polyhedra are notoriously difficult objects to handle from a theoretical standpoint. One reason for this is that algorithms are designed to handle worst-case objects, which tend to have pointy spikes or sharp folds that do not occur in realistic inputs. Some, but not all of this difficulty is alleviated by developing algorithms for fat polyhedra. The PI proposes to study the complexity of several problems on polyhedra under the following new input model.

Definition: A polyhedron with triangular facets is **(α, β)-smooth** if and only if every face angle is at least α and every dihedral angle is at least β. A polyhedron is (α, β)-smooth if its boundary has a (α, β)-smooth triangulation without Steiner points. Finally, a family of polyhedra is smooth if every polyhedron is (α, β)-smooth for some constants $0 < \alpha < \pi/3$ and $0 < \beta < \pi$.

This definition is motivated by several different methods used to create polyhedral models in practice. Given a sufficiently dense sample of a surface Σ, recent algorithms of Amenta and others [27, 28, 29, 30] output a smooth polyhedron homeomorphic to Σ. Many polygonalization algorithms for smooth surfaces such as Bézier and NURBS patches [50, 145, 146, 151, 173] also tend to output smooth polyhedra. Smooth polyhedra are also well-shaped surface meshes suitable for numerically stable finite element analysis [194]. A huge number of algorithms have been developed for generating and improving surface meshes; see [49, 167, 194, 205] for recent surveys.

Smooth polyhedra can have non-convex and multiply-connected facets, but every facet of a smooth polyhedron must be fat. (A nonconvex polygon P is fat if, for any disk centered inside P that does not contain P, a constant fraction of its area lies inside P [190].) A smooth boundary triangulation

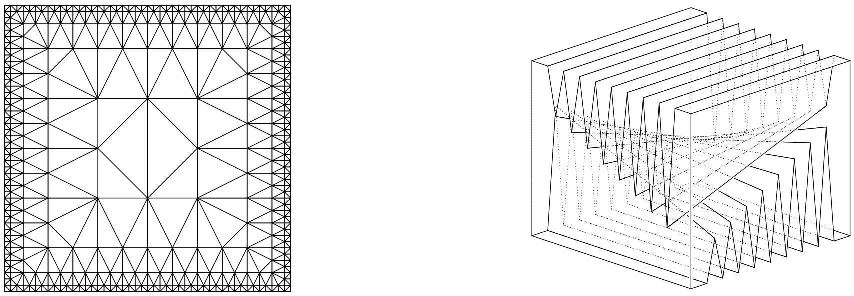

Figure 2. Left: One side of a smoothly triangulated 'pizza box'. Right: Chazelle's polyhedron.

can be computed in linear time for any smooth polyhedron, by constructing the constrained Delaunay triangulation of each facet [73, 75]. Although every facet of a smooth polyhedron must be fat, smooth polyhedra need not be fat themselves, nor must fat polyhedra be smooth. In fact, *any* three-dimensional object can be approximated arbitrarily closely by a smooth polyhedron. For example, an $n \times n \times 1$ 'pizza box' can be approximated by a smooth polyhedron with $\Theta(n)$ facets by adding a graded mesh to each facet; see Figure 2. Unlike fat objects, which cannot have certain features such as thin spikes and sheets, smooth polyhedra can have arbitrary thin features, provided that those features are paid for by a local increase in combinatorial complexity.

Decomposition. Many geometric problems involving non-convex objects can be solved more efficiently if the objects are first triangulated or decomposed into larger convex pieces. For polygons, this is relatively straightforward [63, 139, 67], but as usual, the situation for polyhedra is more complicated. Triangulating a nonconvex polyhedron may be impossible without the introduction of extra vertices, commonly called *Steiner points*. The simplest example of such a polyhedron, due to Schönhardt [182], is a twisted triangular prism. Even approximating the necessary number of Steiner points is NP-hard [176]. Unlike in the plane, different triangulations of the same polyhedron can have different numbers of tetrahedra, and computing a triangulation of minimum complexity is NP-hard [148], even for convex polytopes [42].

On the algorithmic side, any n-vertex polyhedron can be decomposed into convex pieces of total complexity $O(n^2)$ using a binary space partition tree [168]. Agarwal, Guibas, and the PI [11] tightened this bound slightly to $O(n \log^2 n + k)$, where $k = O(n^2)$ is the number of intersections in a projection of the polyhedron's edges to the xy-plane. Polyhedra with $r \ll n$ reflex edges can be decomposed into $\Theta(n + r^2)$ convex polyhedra of total complexity $\Theta(nr + r^{7/3})$ [70, 131]. Several other heuristics for building binary space partition trees are known, and although they seem to work well in practice, most of them have abysmal worst-case performance.

Chazelle [60] described a polyhedron with $O(n)$ vertices that cannot be broken into fewer than $\Omega(n^2)$ convex pieces, implying that the decomposition algorithms described above are optimal in the worst case. Chazelle's polyhedron has both small dihedral angles and (after boundary triangulation) skinny facets; see Figure 2. A variant of Chazelle's polyhedron, due to Eppstein [89], has large dihedral angles but facets with aspect ratio $\Theta(n)$; another, constructed by the PI, has fat facets but with minimum dihedral angle $\Theta(1/n)$. All these polyhedra contain an approximation of a $O(n) \times O(n)$ grid on the hyperbolic paraboloid $z = xy$. The PI has proved that no smooth polyhedron can contain a twisted grid of this form. This is *not* true of fat polyhedra; all three polyhedra can be made fat by further modification. Since a twisted grid seems to be the only way to force quadratic decompositions, the PI optimistically conjectures that smooth polyhedra can be decomposed into $O(n \operatorname{polylog} n)$ convex pieces, and in particular into $O(n \operatorname{polylog} n)$ tetrahedra.

It is quite easy to construct a smooth polyhedron whose edges project to an $O(n) \times O(n)$ grid on the xy-plane. Thus, a naïve application of vertical decompositions or cylindrical binary space partition trees [11, 14] leads to $\Omega(n^2)$ pieces in the worst case. Perhaps for any smooth polyhedron, a vertical decomposition in *some* direction has $o(n^2)$ pieces, but this seems unlikely. However, there is evidence to suggest that smooth polyhedra have subquadratic BSPs. Consider the special case of *orthogonal* polyhedra, each of whose facets is an axis-parallel rectangle. Paterson and Yao [169] describe an orthogonal polyhedron whose every convex decomposition has $\Omega(n^{3/2})$ pieces. On the other hand, Agarwal *et al.* [12] show that any orthogonal polyhedron whose facets have constant aspect ratio—in other words, any smooth orthogonal polyhedron—has a BSP of size $n2^{O(\sqrt{\log n})}$.

The Schönhardt polyhedron is smooth, so it is impossible to triangulate all smooth polyhedra without using Steiner points. Attaching $\Theta(n)$ Schönhardt polyhedra to a smooth triangulation of the sphere gives us a smooth polyhedron that requires $\Omega(n)$ Steiner points to triangulate. Is this the worst example? Can sufficiently smooth polyhedra be triangulated without any Steiner points?

Besides Chazelle's polyhedron and its relatives, several other lower bound examples for polyhedral decomposition require arbitrarily skinny facets, sharp dihedral angles, or (more commonly) both. These include Ruppert and Seidel's proof that minimizing Steiner points is NP-hard [176], Below *et al.*'s proof that computing optimal triangulations of convex polytopes is NP-hard [42], and Hershberger and Snoeyink's lower bound example for ridge cutting [131]. *All* of these polyhedra are fat. Can these problems be solved more efficiently for smooth polyhedra?

Intersection (collision detection revisited). Another problem that is easy for polygons but hard for polyhedra is deciding whether two objects intersect; this is the static version of the collision detection problem discussed in Section 3. As mentioned earlier, algorithms based on bounding volume hierarchies [38, 114, 133, 141, 157] can be forced to spend $\Omega(n^2)$ time in the worst case. The worst-case example consists of two polyhedra that have edges approximating a twisted grid, as in Chazelle's polyhedron. The fastest known algorithm for deciding whether two non-convex polyhedra intersect runs in time $O(n^{8/5+\varepsilon})$ [170]; see also [181]. For polyhedral terrains, the time bound improves to $O(n^{4/3+\varepsilon})$ [68]. These algorithms employ complex range-searching data structures to avoid directly checking all $\Omega(n^2)$ edge pairs. The PI proved that the polyhedral intersection problem is at least as hard as Hopcroft's problem [93] (see page 7). The main idea of the reduction is to replace each point and line with an arbitrarily thin spike. In light of the PI's $\Omega(n^{4/3})$ lower bound for Hopcroft's problem [95], this means that an algorithm that detects intersections in $o(n^{4/3})$ time is incredibly unlikely.

There appears to be little hope for a truly efficient algorithm for polyhedral intersection or collision detection in the worst case, either in theory or in practice, even for fat polyhedra. But since none of the worst-case examples are smooth, the PI optimistically conjectures that intersections between two smooth polyhedra can be detected in $O(n \operatorname{polylog} n)$ time, probably by a modification of some existing algorithm based on range searching [68, 170, 181]. Assuming an efficient algorithm for this problem can be found, the PI will also consider more difficult problems related to collision detection. For example, can the space between two or more smooth polyhedra be decomposed into few elementary pieces? Can smooth polyhedra be separated by a polyhedral surface with few facets? Can such a decomposition or separator be maintained efficiently as the objects move? Finally, the PI will investigate the efficiency of bounding volume hierarchies of smooth polyhedra—OBB-trees seem to hold the most promise [114].

Related problems. Decomposition and intersection are only two of many geometric problems that are difficult in the worst case and for fat polyhedra, but may be easier for smooth polyhedra. Other problems that the PI will consider include motion planning [191], ray shooting [6, 20, 34, 47, 112, 129, 158, 172], hidden surface removal [15, 19, 43, 44, 46, 47, 98, 117, 138, 160, 171, 185], aspect graphs

[53, 88, 107, 175], medial axis construction [74, 80, 96, 186, 187, 203], and bounded-error approximation [9, 21, 84, 105, 128].

7 Educational activities

The PI believes that education is an exciting and integral component of his academic career, and he is firmly committed to excellence in teaching at all levels. As a member of the theory group at the University of Illinois, the PI teaches the standard sequence of undergraduate theoretical courses, from introductory discrete math to senior-level algorithms, as well as specialized graduate courses in computational geometry. His teaching evaluations are consistently among the best in department. The overall goal of the PI's education activities is to help students develop a 'gut-level' mastery of fundamental mathematical techniques without losing sight of equally important practical issues.

Course development. The computer science department has given the PI a broad license to revise the department's algorithms course, which is required for every student in the department, both graduate and undergraduate. The most significant change was the early introduction and ubiquitous use of randomized algorithms. He also included several relatively recent algorithmic techniques such as backwards analysis [183] and output-sensitive algorithms, and recent results such as scapegoat trees [31, 104], Chan's $O(n \log h)$ "shatter-hull" algorithm [57], and 3SUM-hard problems [103]. The PI was listed in the university's "Incomplete List of Teachers Ranked as Excellent by Their Students" based on his student evaluations for his first offering of the course in Spring 1999 [134], and his videotaped lectures and extensive lecture notes [91] have been used for the last two summers as part of the department's Internet Master of Computer Science program [135]. Next fall, the PI teaches this class for the second time.

In addition to revision of existing courses, developing new courses is also an important part of the PI's education plan. The PI has been developing a very successful graduate course in computational geometry. The course emphasizes core theoretical results and techniques, but it is also tailored each semester to the research interests of the students, many of whom work in areas outside geometry. Last semester's class included students in artificial intelligence, graphics, databases, mobile networking, and scientific computing. In addition to the standard homework assignments, students submitted compelling projects applying geometric techniques to problems in their research areas.

The PI also plans to introduce a geometric algorithms course for advanced undergraduates as a possible alternative to the standard algorithms class. Unlike most traditional theory classes, this course will be driven almost entirely by applications and will have a significant programming component. At the same time, it will be an effective alternative vehicle for teaching important algorithm design and analysis techniques not covered in other courses, in part by letting the students see for themselves the effects of applying good (and bad!) theoretical insights to their programs.

Cross-disciplinary outreach. The PI is also involved in several cross-disciplinary activities within the computer science department, as a vehicle for increasing awareness of geometric techniques among the faculty and students and increasing the PI's opportunities to apply his own geometric expertise. As mentioned in Section 3, the PI is collaborating with robotics researchers at the Beckmann Institute on algorithms for motion planning and manipulation. The PI has been working closely with researchers in the Center for Process Simulation and Design [56]—a DARPA-funded project incorporating computer scientists, mechanical engineers, and mathematicians—on the development of mesh generation algorithms for domains that evolve over time. Besides being a tremendous source of new theoretical research problems, a significant incentive of these efforts is the opportunity for the PI to co-supervise students from diverse academic backgrounds.

References Cited

Publications of the PI are indicated *.

[1] P. K. Agarwal. Partitioning arrangements of lines: II. Applications. *Discrete Comput. Geom.* 5:533–573, 1990.

[2] P. K. Agarwal. Ray shooting and other applications of spanning trees with low stabbing number. *SIAM J. Comput.* 21:540–570, 1992.

[3] P. K. Agarwal, L. Arge, G. Brodal, and J. S. Vitter. I/o-efficient dynamic point location in monotone subdivisions. *Proc. 10th ACM-SIAM Sympos. Discrete Algorithms*, pp. 11–20. 1999.

*[4] P. K. Agarwal, L. Arge, and J. Erickson. Indexing moving points. *Proc. 19th Annu. ACM Sympos. Principles Database Syst.*, pp. 175–186. 2000.

*[5] P. K. Agarwal, L. Arge, J. Erickson, P. G. Franciosa, and J. S. Vitter. Efficient searching with linear constraints. *Proc. 17th Annu. ACM Sympos. Principles Database Syst.*, pp. 169–178. 1998.

[6] P. K. Agarwal, B. Aronov, and S. Suri. Stabbing triangulations by lines in 3d. *Proc. 11th Annu. ACM Sympos. Comput. Geom.*, pp. 267–276. ACM Press, June 1995.

[7] P. K. Agarwal, J. Basch, M. de Berg, L. J. Guibas, and J. Hershberger. Lower bounds for kinetic planar subdivisions. *Proc. 15th Annu. ACM Sympos. Comput. Geom.*, pp. 247–254. 1999.

[8] P. K. Agarwal, J. Basch, L. J. Guibas, J. Hershberger, and L. Zhang. Deformable free space tiling for kinetic collision detection. To appear in *Proc. 4th Workshop Algorithmic Found. Robotics*, 2000.

[9] P. K. Agarwal and P. K. Desikan. An approximation algorithm for terrain simplification. *Proc. 8th ACM-SIAM Sympos. Discrete Algorithms*, pp. 139–147. 1997.

*[10] P. K. Agarwal and J. Erickson. Geometric range searching and its relatives. *Advances in Discrete and Computational Geometry*, pp. 1–56. Contemporary Mathematics 223, American Mathematical Society, 1999.

*[11] P. K. Agarwal, J. Erickson, and L. J. Guibas. Kinetic binary space partitions for intersecting segments and disjoint triangles. *Proc. 9th ACM-SIAM Sympos. Discrete Algorithms*, pp. 107–116. 1998.

[12] P. K. Agarwal, E. F. Grove, T. M. Murali, and J. S. Vitter. Binary space partitions for fat rectangles. *SIAM J. Comput.* 29(5):1422–1448, 2000.

[13] P. K. Agarwal, L. J. Guibas, J. Hershberger, and L. Zhang. Deformable free space tiling for collision detection. To appear in *Proc. 4th Workshop Algorithmic Found. Robotics*, 2000.

[14] P. K. Agarwal, L. J. Guibas, T. M. Murali, and J. S. Vitter. Cylindrical static and kinetic binary space partitions. *Proc. 13th Annu. ACM Sympos. Comput. Geom.*, pp. 39–48. 1997.

[15] P. K. Agarwal, M. Katz, and M. Sharir. Computing depth orders for fat objects and related problems. *Comput. Geom. Theory Appl.* 5:187–206, 1995.

[16] P. K. Agarwal and M. van Kreveld. Implicit point location in arrangements of line segments, with an application to motion planning. *Internat. J. Comput. Geom. Appl.* 4:369–383, 1994.

[17] P. K. Agarwal and J. Matoušek. Ray shooting and parametric search. *SIAM J. Comput.* 22(4):794–806, 1993.

[18] P. K. Agarwal and J. Matoušek. Dynamic half-space range reporting and its applications. *Algorithmica* 13:325–345, 1995.

[19] P. K. Agarwal and M. Sharir. Applications of a new space-partitioning technique. *Discrete Comput. Geom.* 9:11–38, 1993.

[20] P. K. Agarwal and M. Sharir. Ray shooting amidst convex polyhedra and polyhedral terrains in three dimensions. *SIAM J. Comput.* 25:100–116, 1996.

[21] P. K. Agarwal and S. Suri. Surface approximation and geometric partitions. *Proc. 5th ACM-SIAM Sympos. Discrete Algorithms*, pp. 24–33. 1994.

[22] A. Aggarwal and J. S. Vitter. The input/output complexity of sorting and related problems. *Commun. ACM* 31:1116–1127, 1988.

[23] M. E. Agishtein and A. A. Migdal. Smooth surface reconstruction from scattered data points. *Comput. Graph. (UK)* 15(1):29–39, 1991.

[24] O. Aichholzer, D. Alberts, F. Aurenhammer, and B. Gärtner. A novel type of skeleton for polygons. *J. Universal Comput. Sci.* 1(12):752–761, 1995. ⟨http://www.iicm.edu/jucs_1_12/a_novel_type_of⟩.

[25] G. Albers, L. J. Guibas, J. S. B. Mitchell, and T. Roos. Voronoi diagrams of moving points. *Internat. J. Comput. Geom. Appl.* 8:365–380, 1998.

[26] P. Amburn, E. Grant, and T. Whitted. Managing geometric complexity with enhanced procedural models. *Comput. Graph* 20(4):189–195, 1986. Proc. SIGGRAPH '86.

[27] N. Amenta and M. Bern. Surface reconstruction by Voronoi filtering. *Discrete Comput. Geom.* 22(4):481–504, 1999.

[28] N. Amenta, M. Bern, and D. Eppstein. The crust and the β-skeleton: Combinatorial curve reconstruction. *Graph. Models Image Process.* 60:125–135, 1998.

[29] N. Amenta, M. Bern, and M. Kamvysselis. A new Voronoi-based surface reconstruction algorithm. *Proc. SIGGRAPH '98*, pp. 415–412. Computer Graphics Proceedings, Annual Conference Series, July 1998.

[30] N. Amenta, S. Choi, T. K. Dey, and N. Leekha. A simple algorithm for homeomorphic surface reconstruction. *Proc. 16th Annu. ACM Sympos. Comput Geom.*, pp. 213–222. 2000.

[31] A. Andersson. General balanced trees. *J. Algorithms* 30:1–28, 1999.

[32] L. Arge, V. Samoladas, and J. S. Vitter. On two-dimensional indexability and optimal range search indexing. *Proc. 18th Annu. ACM Sympos. Principles Database Syst.*, pp. 346–357. 1999.

[33] L. Arge and J. Vahrenhold. I/O-efficient dynamic planar point location. *Proc. 16th Annu. ACM Sympos. Comput. Geom.*, pp. 191–200. 2000.

[34] B. Aronov and S. Fortune. Average-case ray shooting and minimum weight triangulations. *Proc. 13th Annu. ACM Sympos. Comput. Geom.*, pp. 203–211. 1997.

[35] S. Ar, B. Chazelle, and A. Tal. Self-customised BSP trees for collision detection. *Comput. Geom. Theory Appl.* 10:23–29, 2000.

[36] S. Ar and A. Tal. Query-driven BSP trees. *Abstracts 16th European Workshop Comput. Geom.*, pp. 81–84. 2000. ⟨http://www.cs.bgu.ac.il/~cg2000/PS/9.ps⟩.

[37] D. Baraff. Interactive simulation of solid rigid bodies. *IEEE Comput. Graphics Appl.* 15(3):63–75, 1995.

[38] G. Barequet, B. Chazelle, L. Guibas, J. Mitchell, and A. Tal. BOXTREE: A hierarchical representation for surfaces in 3D. *Comput. Graph. Forum* 15(3):C387–C396, C484, Sept. 1996. Proc. Eurographics'96.

*[39] J. Basch, J. Erickson, L. J. Guibas, J. Hershberger, and L. Zhang. Kinetc collision detection for two simple polygons. *Proc. 10th Annu. ACM-SIAM Sympos. Discrete Algorithms*, pp. 102–111. 1999.

[40] J. Basch, L. J. Guibas, and J. Hershberger. Data structures for mobile data. *Proc. 8th ACM-SIAM Sympos. Discrete Algorithms*, pp. 747–756. 1997.

[41] J. Basch, L. J. Guibas, and L. Zhang. Proximity problems on moving points. *Proc. 13th Annu. ACM Sympos. Comput. Geom.*, pp. 344–351. 1997.

[42] A. Below, J. de Loera, and J. Richter-Gebert. Finding minimal triangulations of convex 3-polytopes is NP-hard. *Proc. 11th. Annu. ACM-SIAM Sympos. Discrete Algorithms*, pp. 65–66. 2000. ⟨http://www.inf.ethz.ch/~richter/Papers/mintriang.ps⟩.

[43] M. de Berg. Dynamic output-sensitive hidden surface removal for c-oriented polyhedra. *Comput. Geom. Theory Appl.* 2(3):119–140, 1992.

[44] M. de Berg. *Ray Shooting, Depth Orders and Hidden Surface Removal.* Lecture Notes Comput. Sci. 703. Springer-Verlag, Berlin, Germany, 1993.

[45] M. de Berg. Linear size binary space partitions for fat objects. *Proc. 3rd Annu. European Sympos. Algorithms*, pp. 252–263. Lecture Notes Comput. Sci. 979, Springer-Verlag, 1995.

[46] M. de Berg. Generalized hidden surface removal. *Comput. Geom. Theory Appl.* 5:249–276, 1996.

[47] M. de Berg, D. Halperin, M. Overmars, J. Snoeyink, and M. van Kreveld. Efficient ray shooting and hidden surface removal. *Algorithmica* 12:30–53, 1994.

[48] M. de Berg, M. J. Katz, A. F. van der Stappen, and J. Vleugels. Realistic input models for geometric algorithms. *Proc. 13th Annu. ACM Sympos. Comput. Geom.*, pp. 294–303. 1997.

[49] M. Bern and P. Plassmann. Mesh generation. *Handbook of Computational Geometry*, pp. 291–332. Elsevier Science Publishers B.V. North-Holland, 2000.

[50] J. Bloomenthal. An implicit surface polygonizer. *Graphics Gems IV*, pp. 324–349. Academic Press, 1994.

[51] J.-D. Boissonnat. Shape reconstruction from planar cross-sections. *Comput. Vision Graph. Image Process.* 44(1):1–29, Oct. 1988.

[52] J.-D. Boissonnat and F. Cazals. Smooth surface reconstruction via natural neighbour interpolation of distance functions. *Proc. 16th Annu. ACM Sympos. Comput. Geom.* 2000.

[53] K. W. Bowyer and C. R. Dyer. Aspect graphs: An introduction and survey of recent results. *Int. J. of Imaging Systems and Technology* 2:315–328, 1990.

[54] G. S. Brodal and R. Jakob. Dynamic planar convex hull with optimal query time and $O(\log n \cdot \log\log n)$ update time. *Proc. 7th Scandinavian Workshop on Algorithm Theory*, p. to appear. Lect. Notes Comput. Sci., Spring-Verlag, 2000.

[55] S. Cameron. Enhancing GJK: Computing minimum and penetration distance between convex polyhedra. *Proc. Internat. Conf. Robotics and Automation*, pp. 3112–3117. 1997.

[56] Center for Process Simluation and Design. ⟨http://www.cse.uiuc.edu/cpsd/⟩.

[57] T. M. Chan. Optimal output-sensitive convex hull algorithms in two and three dimensions. *Discrete Comput. Geom.* 16:361–368, 1996.

[58] T. M. Chan. Dynamic planar convex hull operations in near-logarithmic amortized time. *Proc. 40th Annu. IEEE Sympos. Found. Comput. Sci.*, pp. 92–99. 1999.

[59] T. M. Chan, J. Snoeyink, and C. K. Yap. Primal dividing and dual pruning: Output-sensitive construction of 4-d polytopes and 3-d Voronoi diagrams. *Discrete Comput. Geom.* 18:433–454, 1997.

[60] B. Chazelle. Convex partitions of polyhedra: a lower bound and worst-case optimal algorithm. *SIAM J. Comput.* 13:488–507, 1984.

[61] B. Chazelle. Fast searching in a real algebraic manifold with applications to geometric complexity. *Proc. Internat. Colloq. on Trees in Algebra and Program.*, pp. 145–156. Lecture Notes Comput. Sci. 185, Springer-Verlag, 1985.

[62] B. Chazelle. Lower bounds on the complexity of polytope range searching. *J. Amer. Math. Soc.* 2:637–666, 1989.

[63] B. Chazelle. Triangulating a simple polygon in linear time. *Discrete Comput. Geom.* 6(5):485–524, 1991.

[64] B. Chazelle. Lower bounds for off-line range searching. *Discrete Comput. Geom.* 17(1):53–66, 1997.

[65] B. Chazelle. A spectral approach to lower bounds with applications to geometric searching. *SIAM J. Comput.* 27(2):545–556, 1998.

[66] B. Chazelle. Geometric complexity and the discrepancy method. *Abstracts 15th European Workshop Comput. Geom.*, pp. 21–23. INRIA Sophia-Antipolis, 1999.

[67] B. Chazelle and D. P. Dobkin. Optimal convex decompositions. *Computational Geometry*, pp. 63–133. North-Holland, 1985.

[68] B. Chazelle, H. Edelsbrunner, L. J. Guibas, and M. Sharir. Algorithms for bichromatic line segment problems and polyhedral terrains. *Algorithmica* 11:116–132, 1994.

[69] B. Chazelle and A. Lvov. A trace bound for the hereditary discrepancy. *Proc. 17th Annu. ACM Sympos. Comput. Geom.*, pp. 64–69. 2000.

[70] B. Chazelle and L. Palios. Triangulating a non-convex polytope. *Discrete Comput. Geom.* 5:505–526, 1990.

[71] B. Chazelle and B. Rosenberg. Simplex range reporting on a pointer machine. *Comput. Geom. Theory Appl.* 5:237–247, 1996.

[72] B. Chazelle and E. Welzl. Quasi-optimal range searching in spaces of finite VC-dimension. *Discrete Comput. Geom.* 4:467–489, 1989.

[73] L. P. Chew. Constrained Delaunay triangulations. *Algorithmica* 4:97–108, 1989.

[74] F. Chin, J. Snoeyink, and C. A. Wang. Finding the medial axis of a simple polygon in linear time. *Discrete Comput. Geom.* 21(3):405–420, 1999.

[75] F. Chin and C. A. Wang. Finding the constrained Delaunay triangulation and constrained Voronoi diagram of a simple polygon in linear time. *SIAM J. Comput.* 28:471–486, 1998.

[76] J. Chomicki and P. Z. Revesz. A geometric framework for specifying spatiotemporal objects. *Proc. 6th Intl. Workshop on Time Representation and Reasoning*, pp. 41–46. 1999.

[77] K. Chung and W. Weng. Quick collision detection of polytopes in virtual environments. *Proc. 3rd ACM Sympos. Virtual Reality Software and Technology*, pp. 125–132. 1996.

[78] K. L. Clarkson. Nearest neighbors in metric spaces. *Discrete Comput. Geom.* 22:63–93, 1999.

[79] K. L. Clarkson and P. W. Shor. Applications of random sampling in computational geometry, II. *Discrete Comput. Geom.* 4:387–421, 1989.

[80] O. Devillers. Randomization yields simple $O(n \log^* n)$ algorithms for difficult $\Omega(n)$ problems. *Internat. J. Comput. Geom. Appl.* 2(1):97–111, 1992. ⟨http://www-sop.inria.fr/prisme/publis/d-rysoa-92.ps.gz⟩.

[81] D. Dobkin, J. Hershberger, D. Kirkpatrick, and S. Suri. Implicitly searching convolutions and computing depth of collision. *Proc. 1st Annu. SIGAL Internat. Sympos. Algorithms*, pp. 165–180. Lecture Notes Comput. Sci. 450, Springer-Verlag, 1990.

[82] D. P. Dobkin and D. G. Kirkpatrick. Fast detection of polyhedral intersection. *Theoret. Comput. Sci.* 27(3):241–253, Dec. 1983.

[83] J. R. Driscoll, N. Sarnak, D. D. Sleator, and R. E. Tarjan. Making data structures persistent. *J. Comput. Syst. Sci.* 38:86–124, 1989.

[84] R. M. Dudley. Metric entropy of some classes of sets with differentiable boundaries. *J. Approx. Theory* 10:227–236, 1974.

[85] R. A. Dwyer. Higher-dimensional Voronoi diagrams in linear expected time. *Discrete Comput. Geom.* 6:343–367, 1991.

[86] H. Edelsbrunner and N. R. Shah. Incremental topological flipping works for regular triangulations. *Algorithmica* 15:223–241, 1996.

[87] A. Efrat and M. Sharir. On the complexity of the union of fat objects in the plane. *Proc. 13th Annu. ACM Sympos. Comput. Geom.*, pp. 104–112. 1997.

[88] D. Eggert, L. Stark, and K. Bowyer. Aspect graphs and their use in object recognition. *Ann. Math. Artif. Intell.* 13:347–376, 1995.

[89] D. Eppstein. Three untetrahedralizable objects, 1997. ⟨http://www.ics.uci.edu/~eppstein/junkyard/untetra/⟩.

*[90] D. Eppstein and J. Erickson. Raising roofs, crashing cycles, and playing pool: Applications of a data structure for finding pairwise interactions. *Discrete Comput. Geom.* 22(4):569–592, 1999.

*[91] J. Erickson. CS 373: Combinatorial Algorithms (Spring 1999). ⟨http://www.uiuc.edu/~jeffe/teaching/373/s99/⟩.

*[92] J. Erickson. Lower bounds for linear satisfiability problems. *Proc. 6th ACM-SIAM Sympos. Discrete Algorithms*, pp. 388–395. 1995. Full version to appear in *Chicago J. Theor. Comput. Sci.*, ⟨http://www.uiuc.edu/~jeffe/pubs/linsat.html⟩.

*[93] J. Erickson. On the relative complexities of some geometric problems. *Proc. 7th Canad. Conf. Comput. Geom.*, pp. 85–90. 1995. ⟨http://www.uiuc.edu/~jeffe/pubs/relative.html⟩.

*[94] J. Erickson. *Lower Bounds for Fundamental Geometric Problems.* Ph.D. thesis, Comput. Sci. Div., Univ. of California, Berkeley, CA, July 1996.

*[95] J. Erickson. New lower bounds for Hopcroft's problem. *Discrete Comput. Geom.* 16:389–418, 1996.

*[96] J. Erickson. Medial surfaces of polyhedra and Voronoi diagrams of lines, 1998. ⟨http://www.uiuc.edu/~jeffe/open/medialxis.html⟩.

*[97] J. Erickson. New lower bounds for convex hull problems in odd dimensions. *SIAM J. Comput.* 28:1198–1214, 1999.

*[98] J. Erickson. Finite-resolution hidden surface removal. *Proc. 11th Annu. ACM-SIAM Sympos. Discrete Algorithms*, pp. 901–909. 2000.

*[99] J. Erickson. Space-time tradeoffs for emptiness queries. *SIAM J. Comput.* 29(6):1968–1996, 2000.

*[100] J. Erickson, L. J. Guibas, J. Stolfi, and L. Zhang. Separation-sensitive collision detection for convex objects. *Proc. 10th Annu. ACM-SIAM Sympos. Discrete Algorithms*, pp. 327–336. 1999.

*[101] J. Erickson and R. Seidel. Better lower bounds on detecting affine and spherical degeneracies. *Discrete Comput. Geom.* 13:41–57, 1995. Erratum in *Discrete Comput. Geom.* 13(1):41–57, 1997.

[102] R. Y. Flatland and C. V. Stewart. Extending rectangular range reporting with query sensitive analysis. *Proc. 8th Canad. Conf. Comput. Geom.*, pp. 319–324. Carleton University Press, Ottawa, Canada, 1996. ⟨http://www.cs.rpi.edu/~flatland/pspapers/cccg96.ps⟩.

[103] A. Gajentaan and M. H. Overmars. On a class of $O(n^2)$ problems in computational geometry. *Comput. Geom. Theory Appl.* 5:165–185, 1995.

[104] I. Galperin and R. L. Rivest. Scapegoat trees. *Proc. 4th Annu. ACM-SIAM Sympos. Discrete Algorithms*, pp. 165–174. 1993.

[105] M. Garland and P. S. Heckbert. Surface simplification using quadric error metrics. *Proc. SIGGRAPH '97*, pp. 209–216. 1997.

[106] P.-L. George and H. Borouchaki. *Triangulation de Delaunay et maillage. Applications aux éléments finis.* Hermes, Paris, France, 1997.

[107] Z. Gigus, J. Canny, and R. Seidel. Efficiently computing and representing aspect graphs of polyhedral objects. *IEEE Trans. Pattern Anal. Mach. Intell.* 13(6):542–551, June 1991.

[108] E. G. Gilbert, D. W. Johnson, and S. S. Keerthi. A fast procedure for computing the distance between complex objects. *Internat. J. Robot. Autom.* 4(2):193–203, 1988.

[109] C. Gold. Crust and anti-crust: A one-step boundary and skeleton extraction algorithm. *Proc. 15th Annu. ACM Sympos. Comput. Geom.*, pp. 189–196. 1999.

[110] M. Golin and H. Na. On the average complexity of 3D-Voronoi diagrams of random points on convex polytopes. To appear in *Proc. 12th Canad. Conf. Comput. Geom.*, 2000.

[111] M. T. Goodrich. A polygonal approach to hidden-line and hidden-surface elimination. *CVGIP: Graph. Models Image Process.* 54(1):1–12, 1992.

[112] M. T. Goodrich. An improved ray shooting method for constructive solid geometry models via tree contraction. *Internat. J. Comput. Geom. Appl.* 7:1–24, 1998.

[113] M. T. Goodrich, J.-J. Tsay, D. E. Vengroff, and J. S. Vitter. External-memory computational geometry. *Proc. 34th Annu. IEEE Sympos. Found. Comput. Sci.*, pp. 714–723. 1993.

[114] S. Gottschalk, M. C. Lin, and D. Manocha. OBB-tree: A hierarchical structure for rapid interference detection. *Comput. Graph.* ??:171–180, 1996. Proc. SIGGRAPH '96.

[115] R. Grossi and G. F. Italiano. Efficient cross-tree for external memory. *External Memory Algorithms*, pp. 87–106. DIMACS Series in Discrete Mathematics and Theoretical Computer Science 50, American Mathematical Society, 1999.

[116] R. Grossi and G. F. Italiano. Efficient splitting and merging algorithms for order decomposable problems. *Inform. Comput.* 154(1):1–33, 1999.

[117] E. F. Grove, T. M. Murali, and J. S. Vitter. The object complexity model for hidden-surface removal. *Internat. J. Comput. Geom. Appl.* 9(2):207–217, Apr. 1999.

[118] L. J. Guibas. Kinetic data structures — a state of the art report. *Proc. Workshop Algorithmic Found. Robot.*, pp. 191–209. A. K. Peters, 1998.

[119] L. J. Guibas, J. Hershberger, S. Suri, and L. Zhang. Kinetic connectivity for unit disks. *Proc. 16th Annu. ACM Sympos. Comput. Geom.*, pp. 331–331. 2000.

[120] L. J. Guibas, D. Hsu, and L. Zhang. H-Walk: Hierarchical distance computation for moving convex bodies. *Proc. 15th Annu. ACM Sympos. Comput. Geom.*, pp. 344–351. 1999.

[121] L. J. Guibas, D. Hsu, and L. Zhang. A heirarchical method for real-time distance computation among moving convex bodies. *Comput. Geom. Theory Appl.* 15(1–3):51–68, 2000.

[122] L. J. Guibas, D. E. Knuth, and M. Sharir. Randomized incremental construction of Delaunay and Voronoi diagrams. *Algorithmica* 7:381–413, 1992.

[123] L. J. Guibas, J. S. B. Mitchell, and T. Roos. Voronoi diagrams of moving points in the plane. *Proc. 17th Internat. Workshop Graph-Theoret. Concepts Comput. Sci.*, pp. 113–125. Lecture Notes Comput. Sci. 570, Springer-Verlag, 1992.

[124] L. J. Guibas, M. H. Overmars, and M. Sharir. Ray shooting, implicit point location, and related queries in arrangements of segments. Report 433, Dept. Comput. Sci., New York Univ., Mar. 1989.

[125] L. J. Guibas, J. Snoeyink, and L. Zhang. Compact Voronoi diagram for moving convex polygons. To appear in *Proc. 7th Scandinadvian Worsh. Algorithm Theory*, 2000.

[126] B. Guo, J. Menon, and B. Willette. Surface reconstruction using alpha shapes. *Comput. Graph. Forum* 16(4):177–190, Oct. 1997.

[127] S. Har-Peled and M. Sharir. On-line point location in planar arrangments and its applications. Manuscript, 1999. ⟨http://www.cs.duke.edu/~sariel/papers/99/run.html⟩.

[128] P. Heckbert and M. Garland. Optimal triangulation and quadric-based surface simplification. *Comput. Geom. Theory Appl.* 14(1–3):49–65, 1999.

[129] J. Hershberger and S. Suri. A pedestrian approach to ray shooting: Shoot a ray, take a walk. *J. Algorithms* 18:403–431, 1995.

[130] J. Hershberger and S. Suri. Kinetic connectivity of rectangles. *Proc. 15th Annu. ACM Sympos. Comput. Geom.*, pp. 237–246. 1999.

[131] J. E. Hershberger and J. S. Snoeyink. Erased arrangements of lines and convex decompositions of polyhedra. *Comput. Geom. Theory Appl.* 9:129–143, 1998.

[132] P. M. Hubbard. Collision detection for interactive graphics applications. *IEEE Trans. Visualization and Computer Graphics* 1(3):218–230, Sept. 1995.

[133] P. M. Hubbard. Approximating polyhedra with spheres for time-critical collision detection. *ACM Trans. Graph.* 15(3):179–210, July 1996.

[134] Incomplete List for Fall 1999. ⟨http://www.oir.uiuc.edu/dme/ices/f99incl.html⟩.

[135] Internet Master of Computer Science program, distance learning program. ⟨http://www.cs.uiuc.edu/education/grad/degrees/imcs/index.html⟩.

[136] M. J. Katz. 3-D vertical ray shooting and 2-D point enclosure, range searching, and arc shooting amidst convex fat objects. *Comput. Geom. Theory Appl.* 8(6):299–316, 1997.

[137] M. J. Katz. *Geometric Optimization via Expanders and Visibility of Fat Objects in Three Dimensions: Two Studies in Computational Geometry*. Ph.D. thesis, Tel-Aviv University, Tel-Aviv, Israel, 1994.

[138] M. J. Katz, M. H. Overmars, and M. Sharir. Efficient hidden surface removal for objects with small union size. *Proc. 7th Annu. ACM Sympos. Comput. Geom.*, pp. 31–40. 1991.

[139] M. Keil and J. Snoeyink. On the time bound for convex decomposition of simple polygons. *Proc. 10th Annu. Canadian Conf. Comput. Geom.* 1998. ⟨http://cgm.cs.mcgill.ca/cccg98/proceedings/cccg98-keil-time.ps.gz⟩. Full version to appear in *Int. J. Comput. Geom. Appl.*, ⟨http://www.cs.unc.edu/~snoeyink/papers/convdecomp.ps.gz⟩.

[140] D. Kirkpatrick, J. Snoeyink, and B. Speckmann. Kinetic collision detection for simple polygons. *Proc. 16th Annu. ACM Sympos. Comput. Geom.*, pp. 321–330. 2000.

[141] J. Klosowski, M. Held, J. S. B. Mitchell, K. Zikan, and H. Sowizral. Efficient collision detection using bounding volume hierarchies of k-DOPs. *IEEE Trans. Visualizat. Comput. Graph.* 4(1):21–36, 1998.

[142] G. Kollios, D. Gunopulos, and V. J. Tsotras. Nearest neighbor queries in a mobile environment. *Spatio-Temporal Database Management*, pp. 119–134. 1999.

[143] G. Kollios, D. Gunopulos, and V. J. Tsotras. On indexing mobile objects. *Proc. 18th Annu. ACM Sympos. Principles Database Syst.*, pp. 261–272. 1999.

[144] M. van Kreveld. On fat partitioning, fat covering, and the union size of polygons. *Comput. Geom. Theory Appl.* 9(4):197–210, 1998.

[145] S. Kumar and D. Manocha. Efficient rendering of trimmed NURBS surfaces. *Comput. Aided Design* 27(7):509–521, July 1995.

[146] S. Kumar, D. Manocha, and A. Lastra. Interactive display of large NURBS models. *IEEE Trans. on Visualization and Computer Graphics* 2:323–336, 1996.

[147] D. T. Lee and Y. F. Wu. Geometric complexity of some location problems. *Algorithmica* 1:193–211, 1986.

[148] A. Lingas. The power of non-rectilinear holes. *Proc. 9th Internat. Colloq. Automata Lang. Program.*, pp. 369–383. Lecture Notes Comput. Sci. 140, Springer-Verlag, 1982.

[149] M. C. Lin. *Efficient Collision Detection for Animation and Robotics*. Ph.D. thesis, Dept. Elec. Engin. Comput. Sci., Univ. California, Berkeley, CA, 1993.

[150] M. C. Lin and J. F. Canny. Efficient algorithms for incremental distance computation. *Proc. IEEE Internat. Conf. Robot. Autom.*, pp. 1008–1014. vol. 2, 1991.

[151] W. Lorensen and H. Cline. Marching cubes: a high resolution 3d surface construction algorithm. *Comput. Graph.* 21(4):163–170, 1987.

[152] J. Matoušek. Efficient partition trees. *Discrete Comput. Geom.* 8:315–334, 1992.

[153] J. Matoušek. Range searching with efficient hierarchical cuttings. *Discrete Comput. Geom.* 10(2):157–182, 1993.

[154] J. Matoušek, J. Pach, M. Sharir, S. Sifrony, and E. Welzl. Fat triangles determine linearly many holes. *SIAM J. Comput.* 23:154–169, 1994.

[155] B. Mirtich. *Impulse-based Dynamic Simulation of Rigid Body Systems*. Ph.D. thesis, Comput. Sci. Div., Univ. California, Berkeley, CA, 1996.

[156] B. Mirtich. V-Clip: Fast and robust polyhedral collision detection. Tech. Rep. TR-97-05, Mitsubishi Electrical Research Laboratory, July 1997. ⟨http://www.merl.com/projects/vclip/⟩.

[157] B. Mirtich and J. Canny. Impulse-based dynamic simulation. *The Algorithmic Foundations of Robotics*. A. K. Peters, 1995.

[158] J. S. B. Mitchell, D. M. Mount, and S. Suri. Query-sensitive ray shooting. *Internat. J. Comput. Geom. Appl.* 7(4):317–347, Aug. 1997.

[159] D. M. Mount, N. S. Netanyahu, R. Silverman, and A. Wu. Chromatic nearest neighbour searching: a query sensitive approach. *Proc. 7th Canad. Conf. Comput. Geom.*, pp. 261–266. 1995.

[160] K. Mulmuley. An efficient algorithm for hidden surface removal. *Comput. Graph.* 23(3):379–388, 1989 1989.

[161] J. Nievergelt and P. Widmayer. Spatial data structures: Concepts and design choices. *Handbook of Computational Geometry*, pp. 725–764. Elsevier Science Publishers B.V. North-Holland, 2000.

[162] A. Okabe, B. Boots, and K. Sugihara. *Spatial Tessellations: Concepts and Applications of Voronoi Diagrams*. John Wiley & Sons, Chichester, UK, 1992.

[163] J. Orenstein. A comparison of spatial query processing techniques for native and parameter spaces. *Proc. ACM SIGMOD Conf. on Management of Data*, pp. 343–352. 1990.

[164] M. H. Overmars. Point location in fat subdivisions. *Inform. Process. Lett.* 44:261–265, 1992.

[165] M. H. Overmars and J. van Leeuwen. Maintenance of configurations in the plane. *J. Comput. Syst. Sci.* 23:166–204, 1981.

[166] M. H. Overmars and A. F. van der Stappen. Range searching and point location among fat objects. *J. Algorithms* 21:629–656, 1996.

[167] S. Owen. A survey of unstructured mesh generation technology. *Proc. 7th Internat. Meshing Roundtable*, pp. 239–267. 1998. ⟨http://www.andrew.cmu.edu/user/sowen/survey/⟩.

[168] M. S. Paterson and F. F. Yao. Efficient binary space partitions for hidden-surface removal and solid modeling. *Discrete Comput. Geom.* 5:485–503, 1990.

[169] M. S. Paterson and F. F. Yao. Optimal binary space partitions for orthogonal objects. *J. Algorithms* 13:99–113, 1992.

[170] M. Pellegrini. Ray shooting on triangles in 3-space. *Algorithmica* 9:471–494, 1993.

[171] M. Pellegrini. Repetitive hidden surface removal for polyhedra. *J. Algorithms* 21:80–101, 1996.

[172] M. Pellegrini. Ray shooting and lines in space. *Handbook of Discrete and Computational Geometry*, chapter 32, pp. 599–614. CRC Press LLC, 1997.

[173] J. W. Peterson. Tessellation of NURB surfaces. *Graphics Gems IV*, pp. 286–320. Academic Press, 1994.

[174] D. Pfoser, Y. Theodoidis, and C. S. Jensen. Indexing trajectories of moving point objects. Chorochronos Tech. Rept. CH-99-3, 1999. ⟨http://www.dbnet.ntua.gr/~choros/TRs/99/3/report.ps.gz⟩.

[175] H. Plantinga and C. R. Dyer. Visibility, occlusion, and the aspect graph. *Internat. J. Comput. Vision* 5(2):137–160, 1990.

[176] J. Ruppert and R. Seidel. On the difficulty of triangulating three-dimensional non-convex polyhedra. *Discrete Comput. Geom.* 7:227–253, 1992.

[177] S. Šaltenis, C. S. Jensen, S. T. Leutenegger, and M. A. Lopez. Indexing the positions of continuously moving objects. *Proc. 19th Annu. ACM Sympos. Principles Database Syst.*, pp. 331–334. 2000.

[178] B. Salzberg and V. J. Tsotras. A comparison of access methods for temporal data. TimeCenter Technical Report TR-13, 1997. ⟨http://www.cs.auc.dk/research/DP/tdb/TimeCenter/TimeCenterPublications/TR-13.ps.gz⟩.

[179] H. Samet. *Applications of Spatial Data Structures: Computer Graphics, Image Processing, and GIS.* Addison-Wesley, Reading, MA, 1990.

[180] H. Samet. *The Design and Analysis of Spatial Data Structures.* Addison-Wesley, Reading, MA, 1990.

[181] E. Schömer and C. Thiel. Efficient collision detection for moving polyhedra. *Proc. 11th Annu. ACM Sympos. Comput. Geom.*, pp. 51–60. 1995.

[182] E. Schönhardt. Über die Zerlegung von Dreieckspolyedern in Tetraeder. *Math. Ann.* 98:309–312, 1928.

[183] R. Seidel. Backwards analysis of randomized geometric algorithms. *New Trends in Discrete and Computational Geometry*, pp. 37–68. Algorithms and Combinatorics 10, Springer-Verlag, 1993.

[184] M. I. Shamos. Geometric complexity. *Proc. 7th Annu. ACM Sympos. Theory Comput.*, pp. 224–233. 1975.

[185] M. Sharir and M. Overmars. A simple output-sensitive algorithm for hidden-surface removal. *ACM Trans. Graph.* 11:1–11, 1992.

[186] E. C. Sherbrooke. *3-D Shape Interrogation by Medial Axis Transform.* Ph.D. thesis, Dept. Ocean Engineering, Massachusetts Institute of Technology, Cambridge, Massachusetts, 1995.

[187] E. C. Sherbrooke, N. M. Patrikalakis, and E. Brisson. An algorithm for the medial axis transform of 3d polyhedral solids. *IEEE Trans. Visualizat. Comput. Graph.* 2(1):45–61, Mar. 1996.

[188] A. P. Sistla and O. Wolfson. Temporal conditions and integrity constraints in active database systems. *Proc. 1995 ACM SIGMOD Internat. Conf. Management of Data*, pp. 269–280. 1995.

[189] A. P. Sistla, O. Wolfson, S. Chamberlain, and S. Dao. Modeling and querying moving objects. *Proc. Intl Conf. Data Engineering*, pp. 422–432. 1997.

[190] A. F. van der Stappen. *Motion Planning amidst Fat Obstacles*. Ph.D. dissertation, Dept. Comput. Sci., Utrecht Univ., Utrecht, Netherlands, 1994.

[191] A. F. van der Stappen, D. Halperin, and M. H. Overmars. The complexity of the free space for a robot moving amidst fat obstacles. *Comput. Geom. Theory Appl.* 3:353–373, 1993.

[192] S. Suri, P. M. Hubbard, and J. F. Hughes. Analyzing bounding boxes for object intersection. To appear in *ACM Trans. Graphics*, 2000.

[193] J. Tayeb, O. Ulusoy, and O. Wolfson. A quadtree-based dynamic attribute indexing method. *The Computer Journal* 41:185–200, 1998.

[194] S.-H. Teng and C. W. Tong. Unstructured mesh generation: Theory, practice, and perspectives. To appear in *Internat. J. Comput. Geom. Appl.* ⟨http://www-sal.cs.uiuc.edu/~steng/mesh97_changed.ps⟩.

[195] D. E. Vengroff and J. S. Vitter. Efficient 3-d range searching in external memory. *Proc. 28th Annu. ACM Sympos. Theory Comput.* May 1996.

[196] J. S. Vitter. External memory algorithms and data structures: Dealing with MASSIVE DATA. To appear in *ACM Comput Surv.* ⟨http://www.cs.duke.edu/~jsv/Papers/Vit.IO_survey.ps.gz⟩.

[197] J. Vleugels. *On Fatness and Fitness — Realistic Input Models for Geometric Algorithms*. Ph.D. thesis, Dept. Comput. Sci., Univ. Utrecht, Utrecht, The Netherlands, 1997.

[198] O. Wolfson, S. Chamberlain, L. Jiang, and G. Mendez. Cost and imprecision in modeling the position of moving objects. *Proc. Intl Conf. Data Engineering*, pp. 588–596. 1998.

[199] O. Wolfson, L. Jiang, A. P. Sistla, S. Chamberlain, and M. Deng. Databases for tracking mobile units in real time. *Proc. Int. Conf. Database Theory*, pp. 169–186. 1999.

[200] O. Wolfson, A. P. Sistla, S. Chamberlain, and Y. Yesha. Updating and querying databases that track mobile units. *Distributed and Parallel Databases* 7:257–287, 1999.

[201] O. Wolfson, A. P. Sistla, B. Xu, S. J. Zhou, and S. Chamberlain. DOMINO: Databases for moving objects tracking. *Proc. 1999 ACM SIGMOD Internat. Conf. Management of Data*, pp. 547–549. 1999.

[202] O. Wolfson, B. Xu, S. Chamberlain, and L. Jiang. Moving objects databases: Issues and solutions. *Proc. Sympos. Spatial Database Management*, pp. 111–122. 1998.

[203] X. Yu, J. A. Goldack, and L. Dong. Constructing 3d discrete medial axis. *Proc. ACM Sympos. Solid Modeling Found. and CAD/CAM Applic.*, pp. 481–492. 1991.

[204] G. Zachmann and W. Felger. The BoxTree: Enabling real-time and exact collision detection of arbitrary polyhedra. *Proc. SIVE '95*, pp. 104–113. 1995.

[205] Y. Zheng, R. W. Lewis, and D. T. Gethen. Three-dimensional unstructured mesh generation II: Surface meshes. *Comput. Methods Appl. Mech. Engrg.* 134(3–4):269–284, 1996.

[206] Y. Zhou and S. Suri. Analysis of a bounding box heuristic for collision detection. To appear in *J. ACM.*

[207] Y. Zhou and S. Suri. Collision detection using bounding boxes: Convexity helps. To appear in *Proc. 8th Annu. European Sympos. Algorithms.* Lecture Notes Comput. Sci., Springer-Verlag, 2000.

[208] A. Zomorodian and H. Edelsbrunner. Fast software for box intersection. *Proc. 16th Annu. ACM Sympos. Comput. Geom.*, pp. 129–138. 2000.

Biographical Sketch

Jeff Erickson

Department of Computer Science	(217) 333-6769
University of Illinois at Urbana-Champaign	jeffe@cs.uiuc.edu
1304 W. Springfield Ave.	http://www.uiuc.edu/~jeffe/
Urbana, IL 61801-2987	

Education

1992–1996	University of California, Berkeley: Ph.D. in Computer Science
1990–1992	University of California, Irvine: M.S. in Information and Computer Science
1983–1987	Rice University: B.A. in Computer Science/Mathematical Sciences

Academic employment

Since 1998	Assistant professor, Department of Computer Science, University of Illinois at Urbana-Champaign
1996–1998	Research associate, Center for Geometric Computing, Department of Computer Science, Duke University
1992–1996	Graduate student researcher and graduate student instructor, Computer Science Division, University of California, Berkeley
Fall 1994	Research assistant, Fachbereich Informatik, Universität des Saarlandes, and (unofficially) the Max-Planck-Institut für Informatik, Saarbrücken, Germany
1990–1992	Graduate student researcher and graduate student instructor, Department of Information and Computer Science, University of California, Irvine

Awards and honors

Spring 1999	Incomplete List of Teachers Ranked as Excellent by Their Students, University of Illinois at Urbana-Champaign
1999–2001	Alfred P. Sloan Research Fellowship
1996–1998	NSF Mathematical Sciences Postdoctoral Research Fellowship (DMS-9627683)
1995–1996	Graduate Assistance in Areas of National Need Fellowship

Teaching (all at UIUC)

Fall 1998	CS 497: Geometric Data Structures
Spring 1999	CS 373: Combinatorial Algorithms ⟨http://www.uiuc.edu/~jeffe/teaching/373⟩ • This semester's lecture notes and videotaped lectures were also used in Summer 1999 and Summer 2000 as part of the department's Internet Master of Computer Science program.
Fall 1999	CS 173: Discrete Mathematical Structures
Spring 2000	CS 497: Computational Geometry
Fall 2000	CS 373: Combinatorial Algorithms

Five most relevant publications

[1] Pankaj K. Agarwal, Lars Arge, and *Jeff Erickson*. Indexing moving points. *Proceedings of the 19th ACM Symposium on Principles of Database Systems*, 175–186, 2000. Invited to the special issue of *Journal of Computer and System Sciences* devoted to the conference.

[2] Pankaj K. Agarwal, Lars Arge, *Jeff Erickson*, Paolo Franciosa, and Jeffrey S. Vitter. Efficient searching with linear constraints. *Proceedings of the 17th ACM Symposium on Principles of Database Systems*, 169–178, 1998. Full version to appear in *Journal of Computer and System Sciences*, special issue of invited papers from the 17th Annual ACM Symposium on Principles of Database Systems.

[3] Pankaj K. Agarwal, *Jeff Erickson*, and Leonidas J. Guibas. Kinetic binary space partitions for intersecting segments and disjoint triangles. *Proceedings of the 9th Annual ACM-SIAM Symposium on Discrete Algorithms*, 107–116, 1998.

[4] Julien Basch, *Jeff Erickson*, Leonidas J. Guibas, John Hershberger, and Li Zhang. Kinetic collision detection for two simple polygons. *Proceedings of the 10th Annual ACM-SIAM Symposium on Discrete Algorithms*, 102–111, 1999.

[5] *Jeff Erickson*, Leonidas J. Guibas, Jorge Stolfi, and Li Zhang. Separation-sensitive collision detection for convex objects. *Proceedings of the 10th Annual ACM-SIAM Symposium on Discrete Algorithms*, 327–336, 1999.

Five other publications

[6] Pankaj K. Agarwal and *Jeff Erickson*. Geometric range searching and its relatives. *Advances in Discrete and Computational Geometry*, Bernard Chazelle, Jacob E. Goodman, and Richard Pollack, editors. *Contemporary Mathematics* 223, American Mathematical Society Press, 1999, pp. 1–56.

[7] David Eppstein and *Jeff Erickson*. Raising roofs, crashing cycles, and playing pool: Applications of a data structure for finding pairwise interactions. *Discrete & Computational Geometry* 22(4):569–592, 1999, special issue of invited papers from the 14th Annual ACM Symposium on Computational Geometry.

[8] *Jeff Erickson*. New lower bounds for convex hull problems in odd dimensions. *SIAM Journal on Computing* 28(4): 1198–1214, 1999.

[9] *Jeff Erickson*. New lower bounds for Hopcroft's problem. *Discrete & Computational Geometry* 16(4):389–418, 1996, special issue of invited papers from the 11th Annual ACM Symposium on Computational Geometry.

[10] *Jeff Erickson*. Space-time tradeoffs for emptiness queries. *SIAM Journal on Computing* 29(6):1968–1996, 2000.

Collaborators and other affiliations

Collaborators in previous 48 months. Oswin Aichholzer (Graz, Austria), Lars Arge (Duke), Pankaj K. Agarwal (Duke), Julien Basch (Stanford), Jean-Daniel Boissonnat (INRIA, France), Carmen Cortés (Seville, Spain), Erik D. Demaine (Waterloo, Canada), Vida Dujmović (McGill, Canada), Herbert Edelsbrunner (Illinois and Duke), David Eppstein (U. C. Irvine), Paolo G. Franciosa (Rome, Italy), Leonidas J. Guibas (Stanford), Robert Haber (Illinois), John Hershberger (Mentor Graphics), Henk Meijer (Queen's, Canada), Jean-Pierre Merlet (INRIA, France), Mark Overmars (Utrecht, The Netherlands), Belén Palop (Madrid, Spain), Jean Ponce (Illinois), Suneeta Ramaswami (Rutgers), Alla Sheffer (Illinois), Jorge Stolfi (Campinas, Brazil), John Sullivan (Illinois), Godfried T. Toussaint (McGill, Canada), Jeffrey S. Vitter (Duke), Mariette Yvinec (CNRS and INRIA, France), Li Zhang (Stanford)

Advisors. Raimund Seidel (Saarland, Germany), Ph.D. thesis advisor; Pankaj K. Agarwal (Duke), postdoc sponsor

Graduate students advised or co-advised. Peter Zhe Chu (M.S. expected 2000), Shripad Thite (Ph.D. expected 2003), Steven Zelinka (Ph.D. expected 2003)

Other thesis committees. Tanya Y. Berger-Wolf, Ho-Lun Cheng, André Kündgen (Ph.D. 1999), Cinda Heeren, Peter Leven, Xiang-Yang Li, Alper Üngör, Eric Wang, Afra Zomorodian