Separation Sensitive Kinetic Separation Structures for Convex Polygons

David Kirkpatrick and Bettina Speckmann

Department of Computer Science
University of British Columbia

Abstract. We extend the kinetic data structure for collision detection between moving simple polygons introduced in [14] to incorporate a hierarchical representation of convex chains. This permits us to define and maintain an adaptive hierarchical outer approximation for convex polygons. This representation can be exploited to give separation sensitive complexity bounds for kinetic collision detection comparable to those of Erickson et al. [11] who deal with pairs of convex polygons. More importantly, it forms the basis of a more general representation, developed in a companion paper, that applies to collections of general (not necessarily convex) polygonal objects.

1 Introduction

Collision detection is a basic and unavoidable computational problem arising in all areas of geometric modeling involving objects in motion. Even if we restrict attention to objects in the plane the complexity of collision detection is far from being completely understood.

Consider, for example, the problem of detecting collisions between moving polygonal objects in the plane. Most approaches to collision detection work in two phases. First, a "broad phase" (filtering) algorithm and data structure are used to determine pairs of objects that might possibly collide. A different "narrow phase" (refinement) algorithm and data structure then tests each pair. In general, such approaches force the objects into a representation best suited for one of the two phases or involves some kind of hybrid representation not ideally suited for either.

Recently a number of kinetic data structures (KDS) [4, 13] have been proposed for two-dimensional collision detection. These attempt to avoid arbitrary distinctions between broad and narrow phases and, instead, adapt to changes in the nature and degree of separation between objects. Kinetic data structures exploit the coherence of real motion and maintain, over time, a collection of elementary geometric tests (*certificates*) that together certify that the objects are disjoint. Objects are assumed to have fixed (but changeable) motion trajectories (*flight plans*) and certificates are maintained in a priority queue based on the time of expiration calculated from the current flight plans. When a certificate expires – its test no longer holds – or when a flight plan is changed, the data structure must be updated.

J. Akiyama, M. Kano, and M. Urabe (Eds.): JCDCG 2000, LNCS 2098, pp. 222-236, 2001.

Three natural parameters serve to describe the complexity of a given configuration of objects: k – the number of objects, N – their total number of vertices, and κ – the size of the *minimum link subdivision* [18] separating the objects.

Kinetic data structures and their associated maintenance algorithms can be evaluated and compared with respect to four desired characteristics. A good KDS is *compact* if it uses little space in addition to the input, *responsive* if the data structure invariants can be restored quickly after the failure of a certificate, *local* if it can be updated easily if the flight plan for an object changes, and *efficient* if the worst-case number of events handled by the data structure for a given motion is small compared to some worst-case number of "external events" that must be handled for that motion.

Of particular relevance to the work of this paper are the paper of Basch et al. [3], where pairs of not necessarily convex polygons are maintained using balanced geodesic triangulations, and its extension to collections of polygons by Agarwal et al. [2]. The number of certificates maintained in these schemes is $O(\kappa \log N)$. Furthermore, they have the additional feature that they are canonical (the structures depend only on the current state of the polygons and not on their motion history) which facilitates a non-trivial analysis of their efficiency.

In contrast, the kinetic separation structure (KSS) introduced in [14] maintains a certificate set of size $O(\kappa)$, but is non-canonical (i.e. history dependent) and hence difficult to analyse in terms of efficiency.

Unfortunately, both structures as described are unable to exploit the *degree* of separation of neighboring objects, which is a significant factor in their potential kinetic inefficiency. This shows up, for example, in the fact that the certificate set certifying disjointness of two widely separated translating convex objects (with a total of N vertices) may need to be updated $\Theta(N)$ times, despite the fact that a constant number of certificates would suffice for the entire motion.

One approach to incorporating metric information into kinetic separation structures involves the maintenance of Voronoi diagrams. For example, Guibas et al. [12] study the maintenance of the compact Voronoi diagram of McAllister et al. [16] for a set of disjoint convex polygons moving in the plane. While this structure provides a succinct proof of disjointness, it does so at the expense of a potentially large number of certificate updates even when the objects are widely separated. Furthermore, it has no obvious generalization to collections of non-convex objects.

Another natural approach to exploiting the degree of separation of objects is to approximate each individual object by some kind of coarse outer approximation whose coarseness depends on the separation. The disjointness of the approximations implies the disjointness of the objects but the former may be less expensive to maintain. For isolated convex objects (viewed as the intersection of half spaces determined by their bounding edges) it is possible to construct a hierarchy of successively coarser approximations based on the simple idea of progressive relaxation of half space constraints; a familiar example of this in computational geometry is the Dobkin-Kirkpatrick polyhedral approximation hierarchy [10]. Erickson et al. [11] developed a kinetic collision detection structure

for pairs of *convex* objects based on distance-sensitive variants of such hierarchies, whose performance is thereby made sensitive to the degree of separation between the objects relative to their diameter.

For convex objects in the context of other convex objects or, more generally, for collections of disjoint but possibly interleaved non-convex objects it is not immediately clear what constitutes a useful notion of hierarchical representation or how such a notion could be exploited to provide separation sensitive complexity bounds for kinetic collision detection.

Our objective, in this and a companion paper [15], is to demonstrate that the KSS structure lends itself in a natural way to modifications that provide a measure of sensitivity to the degree (as well as the nature) of the separation of a collection of moving polygons. In this paper we develop the foundation of such a structure by focusing on collections of convex polygons. We exploit hierarchical approximations by replacing convex chains within pseudo-triangles by hierarchical approximation structures. As a consequence, we achieve a reduction in the number of certificate update events as well as more efficient implementations of the pseudo-triangulation update primitives that form the core of the KSS.

Taken together this collection of chain approximations provide an approximation structure for collections of moving convex polygons that is adaptive in the sense that individual polygons are represented at a level of detail sufficient to separate them from their (dynamically changing) set of neighbours. The total representation size (over the entire collection of polygons) is proportional to the number of objects independent of the complexity of the individual objects.

As the objects move the structure shifts its level of detail so as to provide a closer approximation in areas where the polygons are close to each other while coarsening in areas where the polygons are well separated. The approximation is maintained in such a way that it conforms to the partition of the free space into pseudo-triangles. This automatically provides us with the means to detect the parts of the current boundary of the polygons where we can extend the approximation or where we have to increase its level of detail.

A complete analysis of the efficiency of our hierarchical KSS (like that of the unmodified KSS structure) remains a challenge. Nevertheless, it is possible to provide some partial results that demonstrate the potential of this structure. In particular, its performance is comparable with that of the separation sensitive structure of Erickson et al. [11] that was designed for maintaining the separation of *two* convex objects. Specifically, for any constant number of disjoint convex polygons in the plane, let D be their maximum diameter, let N be their total number of vertices, and let s be the minimum distance between all pairs of polygons during their motion. Our structure processes $O(\min\{\log(D/s), \log(N)\})$ events when each of the polygons moves in a linear trajectory. All events can be processed at a cost of $O(\min\{\log(D/s), \log(N)\})$ time per event.

In the next section we recall the important features of the KSS kinetic separation structure. Section 3 sets out the the essential properties of the convex chain approximation hierarchies needed in our separation sensitive augmentation of the KSS, and illustrates several potential representations. Section 4 addresses

the issue of maintaining our augmented KSS, with an emphasis on the dynamic maintenance of the chain approximations that accompany changes in the pseudo-triangulation of the free space and the new kinetic events that accompany the introduction of approximate polygons. Section 5 summarizes the various kinetic attributes of our separation sensitive structure. Finally, section 6 sets out some directions for future work. The Ph.D. dissertation of the second author, currently in preparation, will contain further details on results in this paper and related work.

2 The Kinetic Separation Structure

In this section we revisit the Kinetic Separation Structure (KSS) as it was presented in [14], emphasizing those features that impact its adaptation as a separation sensitive structure. As in [14], we describe how the KSS is built starting from an arbitrary triangulation $\mathcal{T}$ of the free space $\mathcal{E}$ separating a given set of k disjoint simple polygons in the plane together with a point at infinity. This serves to introduces the KSS features in a disciplined fashion. It also makes clear the extent to which the structure is canonical and suggests which features are necessary only for the efficiency (as opposed to the correctness) of the structure.

A KSS can be constructed by first identifying $2k-2$ *partition triangles* in $\mathcal{T}$. Partition triangles are named for the fact that they partition the space between objects (the complement of the polygons and the point at infinity) into disjoint regions called *corridors* (see Fig. 1).

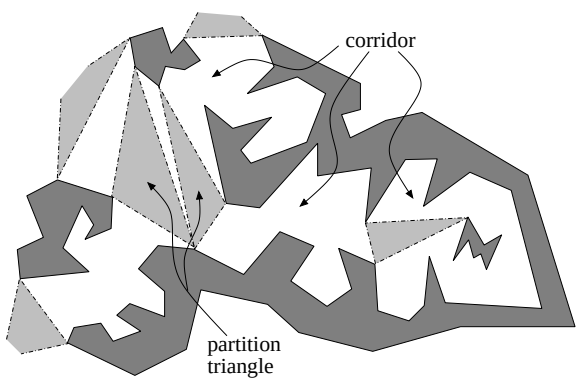

Fig. 1. Decomposition of the free space between polygons.

Lemma 1. *Every corridor is bounded by exactly two edges of partition triangles (*end-edges*) and by exactly two (possibly degenerate) chains of edges of polygons.*

The condition that states that a partition triangle has a positive area is called a *triangle certificate*. Similarly, a *corridor certificate* asserts that none of the polygonal chains or end-edges of a corridor overlap. It is shown that

these certificates suffice to detect all changes in the topological structure of the configuration of polygons.

Theorem 1. *The topological structure of a given configuration of polygons is invariant under motions of the polygons that do not violate any triangle or corridor certificate.*

To establish and maintain corridor certificates a data structure is built inside each corridor that subdivides the empty space between its two defining polygonal chains and connects its two end-edges through a sequence of *pseudo-triangles* (three convex vertices, *corners*, joined by three concave chains).

Consider a triangulated corridor with two bounding polygonal chains labeled γ and δ. Two different kinds of triangulation edges can be distinguished: *bridges* connecting γ with δ and *chords* connecting γ with γ or δ with δ.

The triangles adjacent to bridges and end-edges form a *pathway* as illustrated in Fig. 2. If the pathway is removed from a corridor what remains is a (possibly empty) set of empty pockets, each bounded by a chord (called the *extreme chord* for that pocket). In the dual graph, the edges dual to chords induce a tree (called the *chord tree*) in each pocket (see Fig. 2).

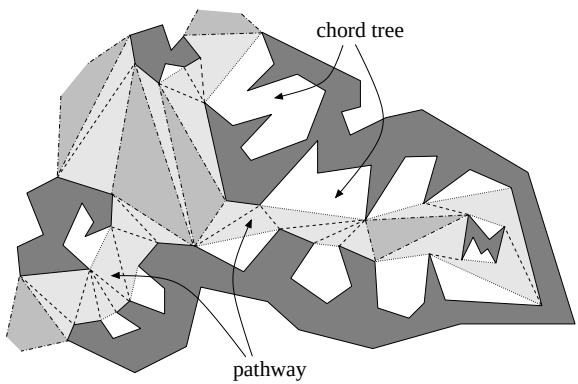

Fig. 2. Decomposition of the corridors into pathways and chord trees.

A pathway is said to be *degenerate* if any of its end-edges or bridges overlap with a chord or polygon edge. Thus to maintain a corridor certificate it suffices to maintain the corresponding non-degenerate pathway. In a similar way the non-degeneracy of a pathway is certified by the non-degeneracy of each of its constituent (pseudo-)triangles.

In order to allow for efficient flight plan updates the set of certificates which at any given time certify the disjointness of objects (the *active set*) should be kept as small as possible. To reduce the size of the active set associated with each corridor the triangles in its associated pathway are merged into a set of pseudo-triangles. The size of the resulting set is linearly related to the size of a minimum link separator for the two polygonal chains defining the corridor.

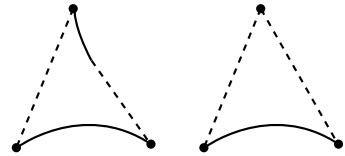

Fig. 3. The two types of object triangles (bridges/end-edges dashed).

This pseudo-triangulation is constructed and maintained by distinguishing two types of (pseudo-)triangles in a given pathway:

- *corner triangles* that have end-edges/bridges on only one side, chords/polygon edges of the same object on the other two; and
- *object triangles* that contain end-edges/bridges on two sides and chords/polygon edges of the same object on the third.

Note that every object triangle in a pathway has at least one side that consists only of a bridge or end-edge (see Fig. 3).

The pseudo-triangles (initially triangles) in a pathway are merged or swapped in order to establish the following invariant:

Invariant 1 (Pathway invariant). *Any three consecutive pseudo-triangles in a pathway must include a corner triangle and any two adjacent object triangles must belong to different objects.*

See Fig. 4 for a fully merged version of the pathways of our example that conforms to the pathway invariant.

The non-degeneracy of a pseudo-triangle can be certified by the non-degeneracy of angles formed at each of its three corners. Thus, the pathway invariant guarantees that the active set for a given corridor has, up to a small additive constant, size proportional to the number of corners in the associated pathway. This is related, in turn, to the size the the minimum link separator for the corridor through the following:

Lemma 2. *A minimum link separator for the two bounding chains of a corridor contains, up to a small additive constant, between one and six times as many segments as there are corner triangles inside the associated pathway.*

Although the number of certificates in the active set is bounded by something close to what could be argued is essential for the certification of disjointness, it is not hard to see that the number or certificate updates required for even very simple motions can vastly exceed what would reasonably be described as necessary. Consider the case of two convex objects with a total of N vertices translating along a straight line (see Fig. 5). As the objects move, each chord and end-edge has to attach to each vertex on each of the polygons. This corresponds to $\Theta(N)$ certificate failures. It seems obvious that a constant number of certificate updates should suffice to certify disjointness in this setting, since the nature of the separation between the objects (i.e. their minimum-link separator) does not

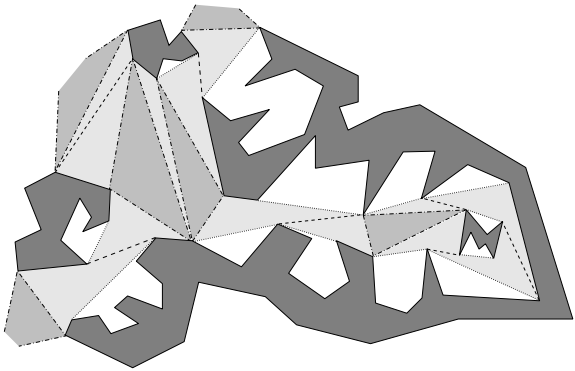

Fig. 4. Minimal pseudo-triangulation for the pathways.

change during the motion and their separation is at all times large with respect to their size.

In the event, of concern in this paper, that all objects are convex, pockets and corner triangles do not exist. It follows from Invariant 1 that every pathway consists of at most two pseudo-triangles, the sides of which contain contiguous fragments of the boundary of at most two objects. In the next section we discuss the construction of dynamic hierarchical representations of these fragments which, in turn, makes possible the formulation separation sensitive bounds on both the responsiveness and efficiency of the structure.

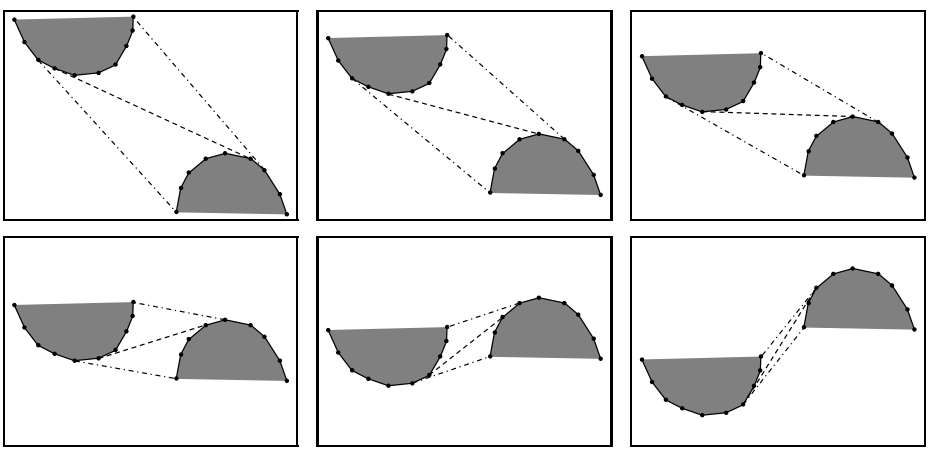

Fig. 5. Two convex objects moving along straight lines.

3 Hierarchical representations of convex chains and relative convex hulls

One of the fundamental reasons for focusing attention on convex objects (either directly or through decomposition) in geometric modeling is the fact that they lend themselves to hierarchical representations. Hierarchical representations of static convex objects permit efficient response to basic intersection (with respect to points or lines) and separation queries [5, 8, 9, 7] and collision detection queries for elementary motions [10]. It is also possible to build hierarchical representations for non-convex objects in terms of their convex pieces to provide responses to separation queries whose cost is a function of the complexity (form) of the separation (measured in terms of the size of the minimal separating chain) [17].

In the context of kinetic collision detection, Erickson et. al [11] described the use of hierarchical representations of individual convex objects to provide separation sensitive maintenance of disjointness certificates for pairs of such objects under a variety of motion models. Critical for this application is additional constraint on outer-hierarchical representations that they be distance sensitive in the sense that the distance of successive levels of the approximation hierarchy from the underlying polygon grows in a controlled fashion that is bounded by some (exponential) function of the level index. This ensures, for example, that two convex objects of diameter at most D and with separation s admit disjoint approximations within their respective hierarchies totalling at most $O(\log(D/s))$ edges.

The hierarchical representations (so-called boomerang hierarchies) proposed by Erickson et. al depend on both the convexity and context independence of the individual objects. Our goal is to bring the benefits of hierarchical representations to a richer context in which we have collections of convex objects in motion. In this setting the hierarchical representations of individual objects must be context dependent and malleable.

We achieve context dependence by building our hierarchical representations in conjunction with the kinetic separation structure (KSS) described in the preceding section. The resulting structure maintains an outer approximation of each object in the collection. Furthermore, at any time the total size of the combined approximations is to within a (small) constant multiple of the total number objects (independent of their complexity).

As was the case with Erickson et. al, we can make use of any hierarchical structure for convex polygonal chains that satisfies certain basic properties. First, we need distance sensitivity as defined above. It will suffice for our present purposes to have a hierarchical representation of chains that is realized as a sequence of $h = \Theta(\log n)$ layers with a total of $\Theta(n)$ edges, where n is the size of the underlying chain (the *base chain*). The first layer is the chain itself and the $(i+1)$-st layer contains at most some fixed fraction of the number of segments of its predecessor. Furthermore, every segment on the $(h-i)$-th layer contains points of distance at most $O(D/2^i)$ from the base chain, where D denotes the diameter of the base chain. The second property that we need is malleability; our approximation for any one polygon is made up of (possibly many) pieces

which may need to be split and appended as the object moves in relation to its neighbours. For our analysis we need a structure that permits a hierarchical representation of a chain to be decomposed into its $O(1)$ top-level subchains in $O(1)$ time. Furthermore, it must support efficient merging of the representations of adjacent co-convex chains (to form a representation of the chain formed by appending the two at their shared endpoint).

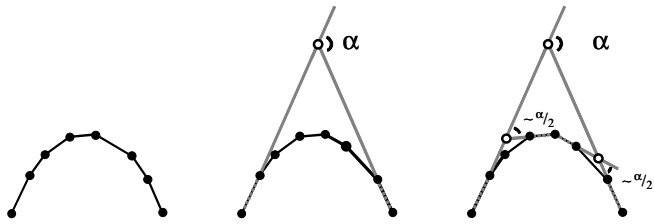

Fig. 6. Constructing the hierarchy.

Any of the boomerang hierarchies proposed by Erickson et. al could be used (or be easily adapted) for our purposes. An even simpler hierarchy is constructed by associating with each vertex in the base chain (except the last) the external angle formed by the extension of its incoming edge and its out-going edge (see Fig. 6). Vertices with angle α greater than $2\pi/n$ are replicated $\lceil \alpha n/(2\pi) \rceil$ times (by the insertion of zero length edges at appropriate angles) so that every (base chain) vertex has angle at most $2\pi/n$.

Since the objects being represented are all convex the relative convex hull of individual objects never changes. This permits an implicit hierarchical representation where the entire boundary of each polygon (after suitable replication of vertices as above) is represented as a simple array. In this case, each base chain is represented by a constant number of pointers (with updates, including merges, in $O(1)$ time) and its hierarchical representation is implemented implicitly using binary search.

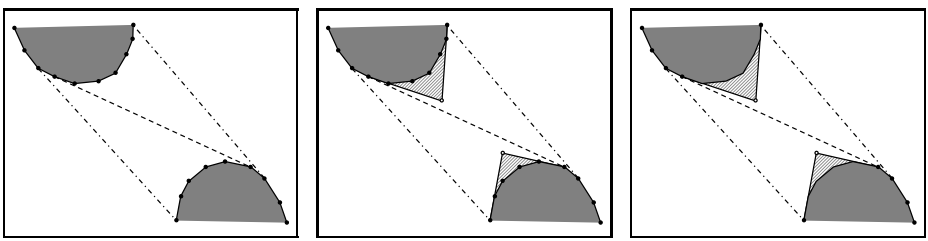

Fig. 7. Creating approximations for the convex chains of the objects in Fig. 5.

We initialize our separation sensitive KSS structure by replacing convex chains in each pseudo-triangle by a hierarchical representation (see Fig. 7). However, this segmentation of the hierarchical representation at pseudo-triangle boundaries is not maintained; doing so would require having every pseudo-triangle corner be a real (as opposed to approximate) polygon vertex, which

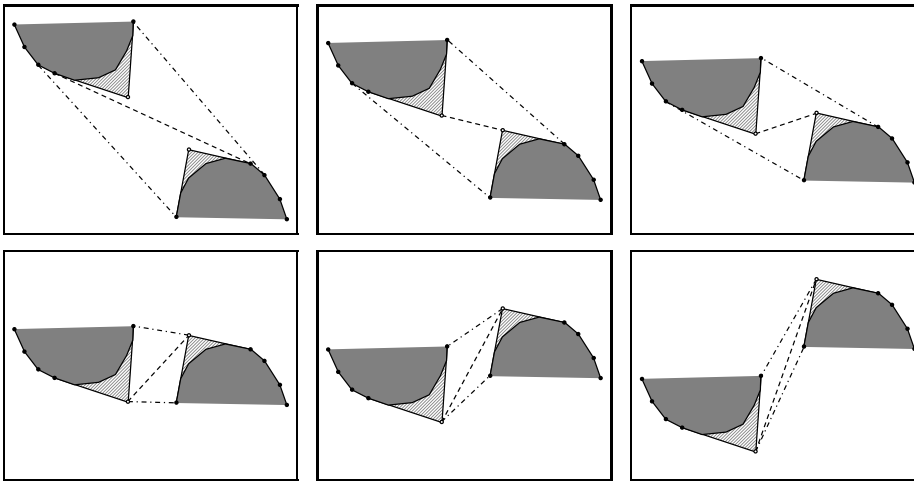

Fig. 8. Two hierarchically represented objects moving along straight lines.

would negate much of the potential benefit of the hierarchical representation. Instead, pseudo-triangles are free to attach at arbitrary approximation vertices subject only to the constraint that the number of approximation chains contained within a single pseudo-triangle is at most some fixed constant. Thus approximation chains are sometimes split, shifted between adjacent pseudo-triangles, and merged, all in $O(1)$ time by suitable adjustment of the pointers into the array structure (see Fig. 9 for a detailed example).

Fig. 8 repeats the motion of the objects of Fig. 5 for two hierarchically represented objects. Note that in this case only a constant number of pseudo-triangle changes occur.

4 Maintenance

We now assume that we are given k convex polygons moving with continuous motion and that we are able to compute the failure time of any certificate associated with those polygons in constant time.

Since we do not distinguish between polygon and approximation vertices as the attachment points of end-edges and bridges all of the KSS update operations as described in [14] can be executed as before. When any certificate fails, we are able to update the data structure and restore the invariants by computation on a constant number of pseudo-triangles. Thanks to the approximation structures relocation of bridges and end-edges and the restoration of invariants can be performed in constant time. As noted earlier, the restructuring of the hierarchies of the pseudo-triangles whenever a complete base chain moves from one triangle into the next can be done in $O(1)$ time by exploiting the implicit nature of our hierarchical representation.

The hierarchy however introduces two new events: (i) a polygon edge or chord collides with an approximation vertex, and (ii) a polygon vertex collides with an edge that ends on an approximation vertex.

Events of type (i) can be handled in $O(1)$ time by "popping" the approximation vertex and restoring the invariants using the convex chain that was previously hidden. See Fig. 9 for an illustration and note how the hierarchy shifts after the collision.

Note that any number of partition triangles can be attached to an approximation vertex whenever it is popped. Since no partition triangle starts out being attached to an approximation vertex the relocation of the partition triangles to a polygon vertex contributes only $O(1)$ amortized cost to this restructuring event.

Events of type (ii) might require multiple approximation vertices (from within the same hierarchy) to be popped in order to determine if a "real" collision occurred. How many layers of the hierarchy have to be traversed is directly proportional to the separation of the objects with respect to their diameters. Specifically, if D is the maximum diameter of the objects and s is their actual object separation at the moment their approximations collide, then $O(\min\{\log(D/s), \log N\})$ layers of the hierarchy will be traversed. Restructuring the hierarchies will again take $O(1)$ time.

5 Kinetic Data Structure Properties

In this section we analyse the properties of our hierarchical KSS structure and highlight the differences with the original structure. Recall that we have k convex polygonal objects consisting of a total of N vertices. Note that the minimum link subdivision for such a collection of objects has size $\Theta(k)$. Our data structure has the following desirable properties for a KDS.

Compactness

There are $\Theta(k)$ partition triangles each of which will contribute a certificate to the active set. Each polygon P contributes $\Theta(\kappa_P)$ certificates, where κ_P is the size of a minimum link cycle separating polygon P from the other objects. Therefore the active set contains $\Theta(k)$ certificates at any time.

Locality

Observe that each polygon is surrounded by a cycle of pathways connected by partition triangles. Due to Lemma 2 we can therefore state that the number of certificates each polygon P appears in is $\Theta(k)$ in the worst case.

Responsiveness

Our data structure uses two kinds of certificates: triangle certificates and corridor (i.e. corner) certificates. Without the hierarchical representation the update of both kinds of certificates can be done in $O(\log n)$ time, where n is the number of vertices of the (at most five) pseudo-triangles involved. With the hierarchical

representation, the corresponding relocation of bridges and end-edges, as well as the restoration of invariants and (implicit) restructuring of the hierarchy, can be accomplished in $O(1)$ time.

In all cases the update involves the destruction and creation of a constant number of edges and associated corner certificates. Each corner and triangle certificate that is disturbed during this process needs to be descheduled or rescheduled in the event queue which takes additional time logarithmic in the size of the active set.

Efficiency

A key performance measure for kinetic data structures is the number of events processed in the worst case. As already pointed out by Agarwal et al. in [1] the notion of efficiency in the case of collision detecting data structures is not clear, since there is no canonical discrete attribute against which to compare the performance of the KDS, i.e. it is not canonically defined what constitutes an external event.

Nevertheless, as was the case with our original structure, it is possible to give an easy upper bound for the number of certificate failures if the polygons translate along algebraic trajectories of bounded degree.

Theorem 2. *If k simple polygons with a total of N vertices translate along algebraic trajectories of degree d then the number of events processed by the KDS is $O(dN^3)$.*

By introducing hierarchical representations of convex chains the efficiency of our structure becomes separation sensitive. In the case of a constant number of disjoint polygons in the plane the efficiency is comparable to the bounds obtained by Erickson et al. in the even more resticted case when the number of convex polygons is restricted to two. Let D be the maximum diameter of all the polygons, let N be their total number of vertices, and let s be the minimum distance between all pairs of polygons during their motion.

Theorem 3. *The number of events processed is $O(\min\{\log(D/s), \log(N)\})$ when each of some constant sized set of polygons moves in a linear trajectory. All events can be processed at a cost of $O(\min\{\log(D/s), \log(N)\})$ time per event.*

Proof. (sketch) Partition triangles and bridges will only attach once to every vertex on a given pathway. Therefore it is sufficient to bound the number of vertices appearing on the pathway (due to a shift in the hierarchy) during the motion. The result follows from the fact that there are at most D/s vertices in the hierarchy at distance at least s from a given polygon. □

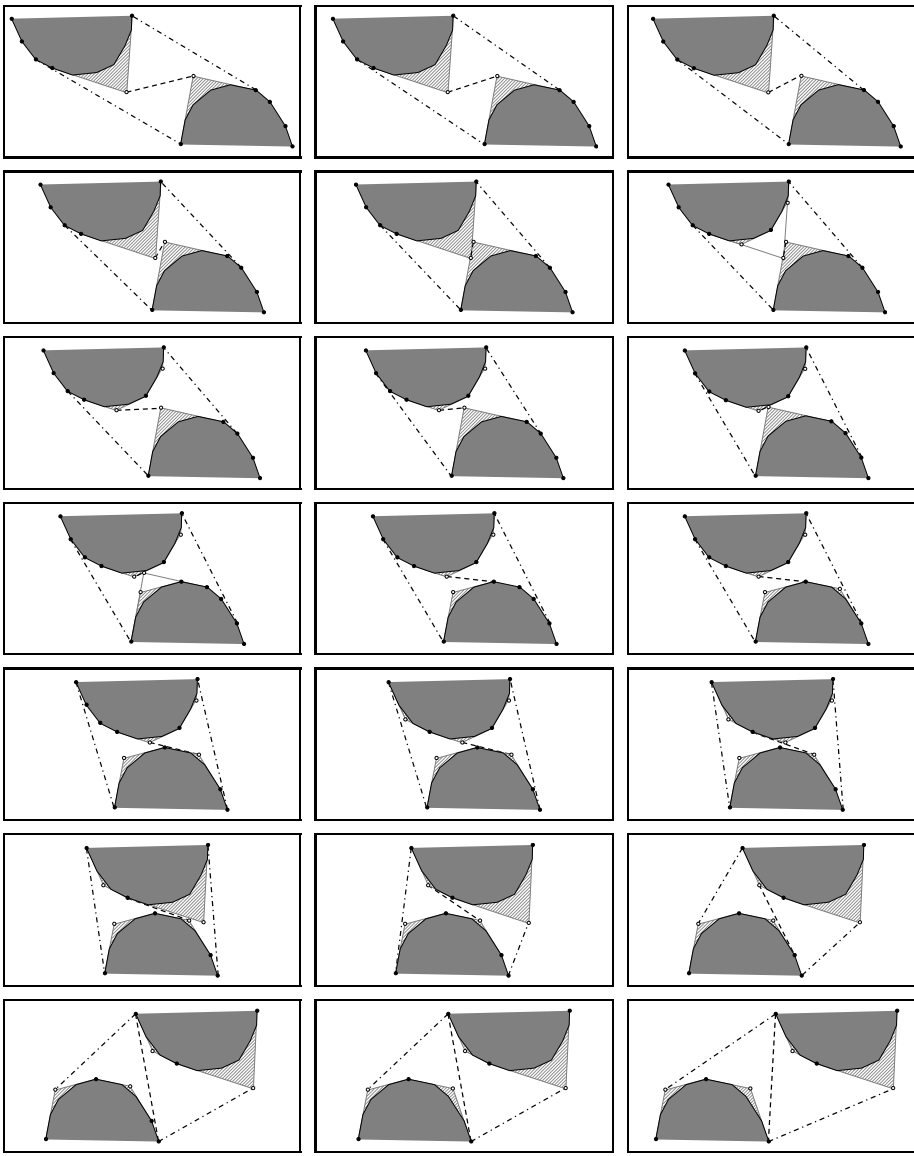

Fig. 9. Flypast of two objects illustrating the adaptive hierarchical representation

6 Future Work

In a companion paper [15] we explore the generalization of the results of this paper to collections of non-convex objects. In addition, we hope to obtain tighter efficiency bounds for various kinds of motions by exploiting the properties of our hierarchical KSS structure. The simplicity of our structure seems promising for extension to two and a half or three dimensions, but the analysis will undoubtedly be even more of a challenge.

References

1. P. K. Agarwal, J. Basch, M. de Berg, L. J. Guibas, and J. Hershberger. Lower bounds for kinetic planar subdivisions. In *Proc. 15th ACM Sympos. Comp. Geom.*, pages 247–254, 1999.
2. P. K. Agarwal, J. Basch, L. J. Guibas, J. Hershberger, and L. Zhang. Deformable free space tilings for kinetic collision detection. In *Proc. 5th Workshop Algorithmic Found. Robotics*, 2000.
3. J. Basch, J. Erickson, L. J. Guibas, J. Hershberger, and L. Zhang. Kinetic collision detection for two simple polygons. In *Proc. 10th ACM-SIAM Sympos. Discrete Algorithms*, pages 102–111, 1999.
4. J. Basch, L. Guibas, and J. Hershberger. Data structures for mobile data. In *Proc. 8th ACM-SIAM Sympos. Discrete Algorithms*, pages 747–756, 1997.
5. B. Chazelle and D. P. Dobkin. Intersection of convex objects in two and three dimensions. *J. ACM*, 34(1):1–27, Jan. 1987.
6. T. H. Cormen, C. E. Leiserson, and R. L. Rivest. *Introduction to Algorithms*. MIT Press, Cambridge, MA, 1990.
7. D. Dobkin, J. Hershberger, D. Kirkpatrick, and S. Suri. Implicitly searching convolutions and computing depth of collision. In *Proc. 1st Annu. SIGAL Internat. Sympos. Algorithms*, volume 450 of *Lecture Notes Comput. Sci.*, pages 165–180. Springer-Verlag, 1990.
8. D. P. Dobkin and D. G. Kirkpatrick. Fast detection of polyhedral intersection. *Theoret. Comput. Sci.*, 27(3):241–253, Dec. 1983.
9. D. P. Dobkin and D. G. Kirkpatrick. A linear algorithm for determining the separation of convex polyhedra. *J. Algorithms*, 6:381–392, 1985.
10. D. P. Dobkin and D. G. Kirkpatrick. Determining the separation of preprocessed polyhedra—A unified approach. In M. S. Paterson, editor, *Automata, Languages and Programming, 17th International Colloquium*, volume 443 of *Lecture Notes in Computer Science*, pages 400–413, Warwick University, England, 16–20 July 1990. Springer-Verlag.
11. J. Erickson, L. Guibas, J. Stolfi, and L. Zhang. Separation-sensitive collision detection for convex objects. In *Proc. 10th ACM-SIAM Sympos. Discrete Algorithms*, pages 327–336, 1999.
12. L. Guibas, J. Snoeyink, and L. Zhang. Compact voronoi diagrams for moving convex polygons. In *To appear Proc. 7th Scandinavian Workshop on Algorithm Theory*, 2000.
13. L. J. Guibas. Kinetic data structures: A state of the art report. In *Proc. 3rd Workshop Algorithmic Found. Robotics*, pages 191–209, 1998.

14. D. Kirkpatrick, J. Snoeyink, and B. Speckmann. Kinetic collision detection for simple polygons. In *Proc. 16th ACM Sympos. on Comp. Geom.*, pages 322–330, 2000.
15. D. Kirkpatrick and B. Speckmann. Separation sensitive kinetic collision detection for simple polygons. In preparation.
16. M. McAllister, D. Kirkpatrick, and J. Snoeyink. A compact piecewise-linear voronoi diagram for convex sites in the plane. *Discrete Comp. Geom.*, 15:73–105, 1996.
17. D. M. Mount. Intersection detection and separators for simple polygons. In *Proc. 8th Annu. ACM Sympos. Comput. Geom.*, pages 303–311, 1992.
18. S. Suri. *Minimum link paths in polygons and related problems.* PhD thesis, Dept. Comp. Sci., Johns Hopkins Univ., 1987.
19. G. T. Toussaint. Shortest path solves translation separability of polygons. Technical Report SOCS-85.27, School Comput. Sci., McGill Univ., 1985.