Notes on Collision Detection

I Survey with (Weller)

1. Scientific work on human perception of physics and the allowed leeway in simuation do exists. Some references are list at [@c1, p.3]

2. Collision detection is a provable difficult problem given the amount of ongoing research.

3. For the polygonal representation (not necessarily convex), the brute force complexity is simply $O(n^2)$ for two objects of n polygons each.

4. One can indeed study average complexity of the above, and this is not an insignificant topic. In fact the author counts deriving such a complexity as a contribution, and states it is $O(n \log n)$.

5. Intuitively, a ***penetration volume*** gives the most information, but it is for now out of reach for interactive applications: "According to Fisher and Lin [7, Sect. 5.1], the penetration volume is "the most complicated yet accurate method" to define the extent of an intersection. However, to our knowledge, there are no algorithms to compute it in real time for a reasonable number of polygons, i.e. more than a dozen of polygons, as yet."

6. The author confirms that finding good benchmarks is not trivial. He points to his own and his advisor's benchmarking approaches, but we still need to verify his claims about its relevance, especially for our pruposes. He says that "a programmer who wants to integrate collision detection into his application still has to choose from hundreds of different approaches. Obviously, this is almost impossible without studying the literature for years. But even for experts it is hard to judge the performance of collision detection algorithms correctly by reading research papers, because almost every researcher presents his results with only certain, well chosen, scenarios. As a remedy, we have developed a standardized benchmarking suite for collision detection algorithms, which we present in Chap. 6. It allows a fair and realistic comparison of different algorithms for a broad spectrum of interesting contact scenarios and many different objects. Moreover, we included a benchmark to compare also the quality of the forces and torques of collision response schemes."

7. ***Chazelle's polyhderon*** is a theoretically very interesting one.

8. From the author's survey of collision detection techniques starting [@c1, p.9], we see that there simply are no (and maybe can be no) especially creative and non-intuitive approaches. The most 'non-intuitive' sub-topic and technique probably being GJK. It seems that any cleverl, computationally complicated approach is doomed for failure for our purposes: "However, even if all these algorithms are close to the optimal solution proved by Edelsbrunner and Maurer [45], in accordance to Zachmann [247], they are profitable over the brute-force method only in scenarios with more than 100 dynamically sim- ulated objects. This is due to the high constant factor that is hidden in the asymptotic notation. Maybe this is also why much more research is done on the acceleration of the narrow phase."

9. For the intersection of n AABBs, it seems that on has indeed found [@c1, p.22] the complexity of the optimal algorithm: $O(n \text{ log2 } n + k)$, which is interesting. Good surveys of similar lower bounds are Mount's [@c2] and [@c3]. The first survey leads to the seemingly very basic result of ***optimal algorithm for intersecting n segments***, which is actually specifically treated in [@c4],[@c5] and [@c6]. The optimal algorithm is of complexity $O(n \log n + k)$. Especially ***Chazelle*** in [@c5] has a very dialectical style; he explictly explains he will give the long-winded and trial-and-error explanation of his results, including a justification of the bound just mentioned.

10. The author mentions attempts at FPGA implementations of some collision detection techniques and comments that they cannot bypass the bounds set by complexity. We note that all these bounds are based on fully random input. This means that opportunities to do better will usually be tied to the existence of extraneous information (such as coherence). We also remark that it is important to have the complexities of available algorithms at hand; the effectiveness of techniques such as broadphase culling are directly related to the fact that doing seemingly more computation, allows saving some amount of computatio at the narrowphase stage, but this only holds if the arithmetic holds up: the the cost of the extra work is less than the cost of what was saved. All things being equal, there is formally nothing 'obvious' about the experimentally verified 'fact' that a broadphase stage is advantageous.

11. Spatial and temporal coherence exploitation are subsumed under the name ***kinetic data structures (KDS)*** in the literature, and it is a topic of research, with a caveat ("all these approaches are limited to two dimensions or very simple objects.") that the author wishes to eliminate. This is explained starting [@c1, p.49].

12. According to [@c8], there are four approaches to CCD:

1. Algebraic Equation Solving

2. Swept Volumes (SV)

3. Kinetic Data Structures

4. Adaptive Subdivision

13. The essence of ***KDS***s can be summarized with: "Usually, a KDS consists of a set of elementary conditions, called certificates, which prove altogether the correctness of its attribute. Those certificates can fail as a result of the motion of the objects. These certificate failures, the so-called events, are placed in an event-queue, ordered according to their earliest failure time. If the attribute changes at the time of an event, the event is called external, otherwise the event is called internal. Thus sampling of time is not fixed but determined by the failure of certain conditions."
A specific example is the auhor gives the examples of points in a rectangle, the points having positions that change with time, and the external events are triggered when it is known that a point leaves the rectangle, while internal events are triggered when the ordering of points within the rectangle changes. A ***flightplan*** is a trajectory between two time instants.

14. There are four established quality criteria for a KDS:

- compact: requires only little space.

- responsive: we can update it quickly in case of a certificate failure.

- local: one object is involved in not too many events. This guarantees that we can adjust changes in the flightplan of the objects quickly.

- efficient: the overhead of internal events with respect to external events is reasonable.

15. Mercifully, the author does document his failed attempts during the development of his kinetic BoxTree. Unfortunately, the comments there invalidate the usefulness of the quality criteria, and (unsurprisingly) undermine the reliability of complexity as the ultimate performance factor. Of course, the well known problem is the constants. In all three exposed 'dead-ends', the problem is the constants, viz. the problem is invisible in the context of the criteria. One wisdom prevails, reinforcing the caution against complexity when it comes to interactive simulation: DOPs do not cut it despite their tighter fit.

16. The ***kinetic separation list*** deserves later attention.

17. A ***CCD algorithm for triangles in the plane*** is presented in [@c1, p.74], based on [@c7]. From this, we warn that the name given for CCD in the literature could actually be ***Dynamic collision detection*** and not 'contiuous collision detection'.

18. The author explains at [@c1,p.87] that when the ***flightpaths*** are unknown (like in the case of a physics simulation), KDSs face a problem. He then offers a naive plan about how to make it work anyway. Having said that, his idea of solving revolves about finding regions where the objects are guaranteed to say in during the simulation, which is an obvious technique, that Havok uses for both accelerating the broaphase and find all possible contacts. Implementaions of this approach in a simulation setting that work locally, and only consider the body itself to determine its volume are in general prone to errors, since such bodies can have their properties change randomly by collisions with other bodies during the frame.

19. From the above, and from [@c9], we feel that continuous collision detection is a global problem, whose solution is dependent on solving the simulation problem itself, which creates an impossible 'chicken and egg' situation, and can end in ***Zeno*** behavior [@c10, @c11].

Bibliography

Ames, Gregg, Zheng. "Is There Life After Zeno? Taking Executions Past the Breaking (Zeno) Point."

Balaban, Ivan J. "An Optimal Algorithm for Finding Segment Intersections." http://club.pdmi.ras.ru/moodle/file.php/15/Materials/p211-balaban.pdf.

Ben-Or, Michael. "Lower Bounds for Algebraic Computation Trees." https://www.hni.uni-paderborn.de/fileadmin/Fachgruppen/Algorithmen/Lehre/Vorlesungsarchiv/SS2011/Concrete_Complexity_Theory/ACTs_Ben-Or.pdf.

Chazelle, Edelsbrunner. "An Optimal Algorithm for Intersecting Line Segments in the Plane." http://www.akira.ruc.dk/~keld/teaching/algoritmedesign_f03/artikler/12/chazelle92.pdf.

Drumwright. "Avoiding Zeno's Paradox in Impulse-Based Rigid Body Simulation."

Eckstein, Schömer. "Dynamic Collision Detection in Virtual Reality Applications."

Mount, D. M. "Geometric Intersection." https://www.cs.umd.edu/~mount/Papers/crc04-intersect.pdf.

———. "Computational Geometry: Proximity and Location." https://www.cs.umd.edu/~mount/Papers/crc05-prox.pdf.

Nohra. "The 'Time of Impact' Argument."

Redon, Lin, Kim. "Fast Continuous Collision Detection for Articulated Models."

Weller, René. "New Geometric Data Structures for Collision Detection and Haptics."