Collisions and Adaptive Levels of Detail

Contact
John Dingliana
Trinity College Dublin
John.Dingliana@cs.tcd.ie

Carol O'Sullivan
Gareth Bradshaw
Trinity College Dublin

Collision detection, contact modeling, and collision response are vital but inherently expensive features of a physically based animation system. As scene complexity increases, collision handling quickly becomes a major bottleneck in the simulation process. A trade-off between speed and accuracy is often required in order to achieve interactive frame-rates[1]. Our goal in ReACT (Real-time Adaptive Collision Toolkit) is to optimize this trade-off by making simplifications as invisible as possible to the viewer.

Many applications simply address the frame-rate problem through pre-emptive simplification, reducing the complexity of the simulation to a pre-determined "safe" level. However, when complexity changes often over the course of a simulation, such an approach suffers from one of two problems: over simplification of the whole for the sake of relatively few snapshots of high complexity, or a drop in frame rate when the computational workload has been underestimated. We can avoid this through adaptive and interactive simplification of the simulation as it evolves. A popular approach to reducing workload in interactive animation is visibility-based culling, where parts of the scene not inside the visible volume are excluded from normal processing[2]. This is taken a step further in ReACT by applying varying levels of simplification over different regions within the viewable area. Such steps are an improvement, but they do not implicitly guarantee target frame rates if static rules are used to determine the levels of detail for different regions. If, for instance, the higher-priority regions should ever encounter computationally complex situations themselves, then problems similar to those in the pre-emptive simplification approach arise.

Target frame rates can be guaranteed by using time-critical mechanisms for computationally expensive parts of the simulation process. A time-critical (or interruptible) mechanism is halted when a scheduler decides that enough time has been spent on any particular task. For such an approach to work, however, we must ensure that some result is obtained and that some degree of correctness is maintained in the system, regardless of when processing is interrupted. This is achieved by using incremental mechanisms, which generate results of increasing accuracy as more time is spent on processing[3]. One example of this is Hubbard's sphere-tree collision detection system, which is extended in ReACT to return increasingly accurate approximations of contact data for collision response calculations (see Figure 1).

Although an interruptible system may guarantee target frame rates, it does not in itself ensure that the trade-off between accuracy and processing time is optimized. For this, we must incorporate some form of prioritization within the process. Certain events or specific parts of the scene are categorized as being more important and, as a result, given more processing time. Prioritization of the scene is based upon factors related to visibility and perceptibility of approximations within different parts of the scene. In ReACT, for instance, priority can be based on a weighted combination of factors such as eccentricity, occlusion, or projected distance from the user's fixation point determined interactively with the use of an eye-tracker (Figure 2).

References
1. Hubbard, P. M. (1995). Real-time collision detection and time-critical computing. *In Proceedings of the First ACM Workshop on Simulation and Interaction in Virtual Environments,* July 1995, 92-96.
2. Chenny, S. & Forsyth, D. (1997). View-dependent culling of dynamic systems in virtual environments. In *Proceedings 1997 Symposium on Interactive 3D Graphics,* 55-58.
3. Dingliana, J. & O'Sullivan, C. (2000). Graceful degradation of collision handling in physically based animation. In *Computer Graphics Forum,* 19 (3), 239-247.

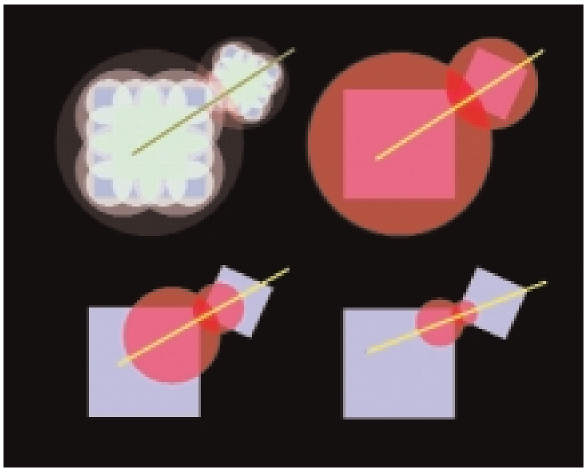
Figure 1. A sphere-tree-based collision-processing system. Shown are the full sphere trees of two objects and the process halted at three different stages. Better approximations of the collision points directly affect the computed response as seen by the direction of the impulse vector in yellow. Red spheres are colliding nodes.

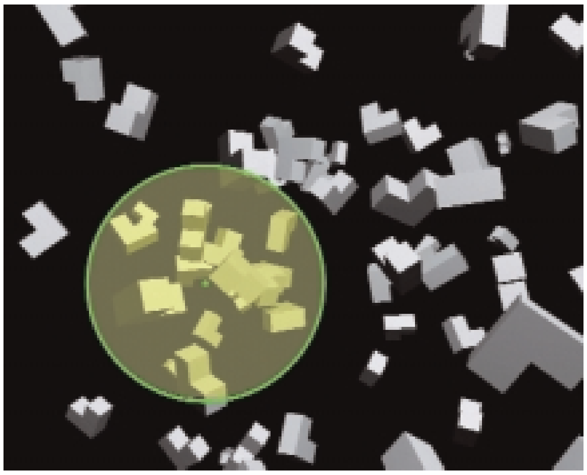
Figure 2. The screenshot shows a projected region on the screen around the user's fixation point, which is given a higher priority, and objects within the region are processed at a higher level of detail.