https://github.com/cs540-25/Methods-in-Collision-Detection.git

Methods in 2D Collision Detection

Benjamin Belandres Alexander Merkle

Department of Electrical Engineering and Computer Science

University of Tennessee, Knoxville

bbelandr@vols.utk.edu amerkle@utk.edu

***Abstract*— Collision detection methods are used in physics engines for a variety of different purposes. While many of these methods have developed into standards for industries such as gaming, the research that has been publicized comparing the strengths and weaknesses of each method is lacking. The goal of this project is to explore these methods so as to increase public understanding of collision detection methods so that one can make an informed decision about which method to employ for their own purposes.**

I. Introduction

Many different kinds of engines use collision detection in order to help simulate physics. Such a wide range of usage leads to a wide quantity of versions that can be used to implement collision detection. This project was born out of interest for the construction of game engines and the mechanics that come together that make them work. The team wants to take a handful of collision detectors and test them against each other in order to produce metrics that can be used to compare them. The idea is not a novel concept, but specifically producing metrics to compare the efficiency of each collision implementation is not explored much in academic papers. Out of the team, Benjamin has had the most experience with engine development, but both members of the team have had prior experience developing programs that use collision detection as well as general experience working with engines.

A. Research Value

This research is primarily intended for a practitioner in the field to use. It will be particularly useful when trying to decide which form of collision detection is the right kind to use for a given scenario. Hopefully, through its publication, the research will help improve the efficiency of individualized services created by people who employ collision by allowing them to make informed decisions.

II. Method

In order to collect the information needed, the team will be building a physics engine capable of switching between and analyzing multiple collision detection methods. Those methods will be as follows:

- Center-Radius Axis-Aligned Bounding Boxes (AABB)
- Bounding Circles
- Sweep and Prune AABB
- Uniform Grids
- BSP Trees

The engine must be able to render multiple physics-enabled objects at the same time. For that reason, the engine will be written in C++ along with the Simple DirectMedia Layer library (SDL) to allow the use of a window for rendering. Everything else will be written from the ground up.

A. Measurements and Performance

To produce metrics, the process's performance will be measured through a ratio of its frames per second and the amount of objects being rendered. Graphics will be made of each method's performance so that they can be easily compared. Tests will be conducted in different scenarios; for instance, there will be a test both for objects

of the same size and for objects of different size.

While on the subject of performance, for now, everything will be single-threaded, but we could explore multithreading as a stretch goal. It would be interesting to see if the amount of threads actually hinders or helps the performance of a certain method of collision detection.

B. Reference Points

The collision will be designed following the guidelines according to Christer Ericson's *Real Time Collision Detection*. Once an implementation is complete, a code review will be conducted with Ericson's examples to ensure that everything was written correctly.

III. Schedule and Project Management

The research can be conducted and completed thanks to the flexibility of the degree to which we can explore collision detection methods. Multiple methods can be used and compared, but limits may be set on the total number we test, as well as on the variety of tests we conduct for each method. Work can best be conducted through in-person meetings, but there is potential for virtual or even asynchronous work.

A. Schedule

We tentatively propose the following schedule of deadlines, with regularly scheduled meetings in the interim periods:

- March 14th - Determination of what software to use; planning and design of implementation of collision detection methods; planning of measurement methods and parameters
- March 28th - Separation and implementation of unique collision detection methods; standardization of tools and assets used to ensure creation of consistent control and experimental tests
- April 11th - Start work with basic proposed tests and record differences between methods for comparison; encounter, track, and flexibly respond to unforeseen obstacles; plan expansion of tests into methods based upon and respectful of software limitations
- April 25th - Completion of tests and data collection
- End of classes - Creation and conclusion of project presentation

B. Risks

Our proposed schedule staggers our necessary steps to an extent that we should be able to complete our research without sacrificing the quality of our conclusions or the thoroughness in which we explore the subject. While the proposed steps in the schedule are all paramount for our process, we do not propose great detail in this schedule so as to make room for unforeseen obstacles. Those obstacles may come in the form of software limitations, collision detection method implementation delays, and troubleshooting for methods that cannot be completed in the planned manner. We should be able to adapt accordingly given the time we have allotted ourselves for each goal.

IV. Team

Both team members have worked directly with game development and the manner within which collision detection is utilized. Benjamin has greater experience in exploring individualized physics engines thanks to past projects at the University of Tennessee, Knoxville. Alec, on the other hand, has experience with scheduling and communications between group members so that the team is on the same page when it comes to individual obligations, goals, and expectations. Ben will take point on choosing software that best fits our goals and methods, as well as mapping out the

implementation of the collision detection methods. Alec can take extensive time for test conduction, ensuring standards are met for test quality and consistency, as well as identifying problem areas that may limit the team's performance or even adjust their methods. Both members will meet regularly to discuss strategies and design and to implement methods by which the research can be completed. Asynchronous work can be employed in the case of testing and data collection, and then the team can reconvene to draw conclusions and design the presentation.