Draft: Object Level Locality in Real-Time Physics Applications

Paul Salzman

Advisor: Professor Glenn Reinman
CS 194 Winter 2007 – Spring 2007

Abstract

As the interactive entertainment industry grows, so does the quest for software realism. Real-time physics simulation has become closely tied with interactive entertainment as it helps to convey a true to life dynamic environment to the user. These simulations require many complex calculations and can utilize computing resources for long periods of time in order to complete. Unfortunately, there is a very strict upper limit on the length of time in which all calculations must complete to allow frames to be rendered at an acceptable rate. In this paper we explore the properties of objects in a physics simulator to see if their correlation to physical objects in motion lends to object level locality and how that can be leveraged for increased performance.

Introduction

Accurate real-time physics simulations are very calculation intensive. In order for such simulations to be practical for use in commercial applications and in interactive entertainment, the physics simulation iterations must complete in a short period of time. This restriction must be enforced to allow other portions of the application to complete and still have processing time for frames to be rendered at an acceptable rate. Some of the more demanding aspects of physics simulation include collision detection, cloth deformation, and fluids. If the values pertaining to the software objects representing physical object properties can be accurately predicted, the gain in performance can be used to increase the complexity and accuracy of these simulations. This would in turn help make the use of real-time physics more feasible for interactive entertainment applications.

One of the more common demanding components in real-time physics simulation is collision detection. Each object in a scene must be compared to all other objects to determine if a collision is occurring. Colliding objects are then grouped together in order to determine which parts of each object's physical representation are colliding and how that affects each objects properties. If object level locality exists, it may be harnessed to help speed up or even predict the result of a collision calculation and may greatly impact the performance of one of the largest portions of the simulator.

Finding methods for improving performance of real-time physics simulations has proved an interesting topic for recent research. Correlation between high level application data locality and critical portions of physical simulation have been found, as well as means of using this locality for improved control prediction [4]. Also, value prediction has been shown to increase process performance when appropriate instructions can be selected receive predicted values [1, 2]. In this paper we plan to explore if object level locality exists in real-time physics simulations and if so, how value prediction can be utilized to increase performance.

The process of examining object level locality will consist of tracing object behavior through selected portions of the ODE. This object trace data must then be analyzed for any appearance of locality. These results will lead to the synthesis of various value predictors that can be tested to take advantage of any locality found at the object level.

Methodology

To observe any object level locality, I will examine object performance in the Open Dynamics Engine (ODE) [3]. ODE is an open source real-time physics simulator that is commonly used in interactive entertainment applications. To fully utilize ODE in an examination of object locality, I will use a performance heavy benchmark created by Thomas Yeh [4] that exercises many facets of physics simulations through multiple concurrent instances of physical object interaction.

The gprof utility will be used to focus in on the functions occupying the large portion of physics simulation time. Upon locating these functions, trace code will be injected into these functions in order to collect data. This code will associate an object with its various states in a chronological order as well as the overall result of the function and dump it into a text file for analysis.

Once the data collection is complete, the traces must be analyzed. The data will be divided into sets defined by distinct objects and functional result values. These subsets will be processed for recent value patterns and stride patterns. These results will be used to determine if there is object level locality. Also using these results, various implementations of value predictors can be created and tested.

Intermediate Results

The function that occupies the most processing time in our benchmark is a box collision function, dBoxBox, which occupies 42.78% of the overall simulation time. The second longest function is also collision detection, collisionAABBs, which occupies 18.06% percent of the simulations processing time. Over 60% of the overall simulation time is dedicated to collision detection, indicating that a boost in collision detection performance can lend to a very large impact in overall performance.

Summary

Not yet available.

References

[1] Brad Calder, Glenn Reinman, and Dean Tullsen. Selective Value Prediction. In *26th International Symposium on Computer Architecture*, May 1999

[2] Mikko Lipasti, Christopher Wilkerson, and John Shen. Value locality and load value prediction. In Seventh International Conference on Architectural Support for Programming Languages and Operating Systems, 1996.

[3] Open Dynamics Engine. http://ode.org/.

[4] Thomas Y. Yeh, Petros Faloutsos, and Glenn Reinman. Accelerating Real-Time Physics Simulation by Leveraging High-Level Information. In *MICRO*, 2006.