Physics Engines: Representing Rigid Bodies in Three Space

Introduction

The modelling of rigid bodies and their motion in three space is a fundamental aspect of several disciplines all over the world. Examples of this technology includes the modelling of structural components in mechanical engineering, the evolution of complex systems in physics computations and the visualization and modeling of cellular interactions in microbiology [1]. In many ways, video game development has paved the way for highly advanced version of this technology to exist by the creation of what are known as "physics engines". Put simply, a physics engine is a piece of software that can approximately simulate physical systems [2]. The first part of this technical review will discuss the basic underlying physical models along with the software and hardware that is required to run physics engines. In the second half, the review will provide a brief overview of the engines that are currently available on the market along with their pricings and known advantages and disadvantages.

Physical Requirements of Physics Engines

A basic physics engine must incorporate most of the following attributes to be able to successfully represent objects in the real world with a reasonable degree of accuracy. In terms of mathematical operations, a physics engine must be capable of performing vector manipulations as well be able to numerically differentiate and integrate in an efficient manner. From a kinematics point of view an engine should be able to simulate linear motion, angular motion in two dimensions, linear momentum, angular momentum, and even allow the use of inertia tensors in both two and three dimensions for more complex physical phenomena [3, 4]. From a static point of view, engines must be able to model Newton's Laws of Motion, field forces, contact forces, stress forces, net forces, moments and net torques [5]. They should also have the ability compute center of masses of each body and the how they may affected under stress and motion vectors. Due to the large number of factors involved, the engines themselves often turn out to be extremely large, complicated pieces of software.

Software and Hardware Requirements of Physics Engines

From a hardware point of view, engines are often very computationally expensive. Parallel programming is a staple part of physics engines and is used substantially to reduce computation time [6]. Nowadays many computers even have separate processing units just to handle the physics involved in the millions of calculations that take place during simulation. These processing units are similar in functionality to GPUs [1]. Thus users must thus have a good understanding of the hardware that is available to them to that they can choose their physics engine accordingly.

From a software point of view, physics engines most generally suffer due to losses in precision in the rendering of their different motion models. Small fluctuations in the data can often results in drastic changes in the predicted outputs with unsatisfactory results. A common complaint with regard to such systems is that the software itself is not very easy to use – often there is a steep learning curve associated with the setting up of engine’s interface before an individual can comfortably begin using an engine for his or her requirements.

Available Physics Engines

There are several physics engines that are currently available on the market. These engines come with several advantages and disadvantages. Often it becomes a matter of trade-offs and the user must decide what system to use based on his or her requirements and needs.

In terms of commercially available physics engines, the most popular one available right now is *Havok* [7]. *Havok* forms the backbone of several of the most successful games ever made including such examples as the Assassin’s Creed franchise. *Havok’s* advantages include that it is a cross-platform product that provides a well-designed, clean and object-oriented C++ API. *Havok* is free to use for all non-commercial developments or applications that are sold for less than $10. A fee is applicable for products that exceed this value, and that fee is determined during the licensing and negotiation involved in the product’s sale [8].

Another commercially available engine is NVIDIA’s *Physx* suite . The *Physx* suite is also highly popular and has the advantage of being free for use for computer-based games. Pricing comes into the picture for games that are developed for consoles [7]. The *Physx* system’s main disadvantage stems from the fact that it has been enabled to use only NVIDIA GPUs for graphics – as a result, a user is restricted to a set of very particular hardware if he or she wishes to use this system.

There is also a plethora of easily available open source and freeware physics engines. These engines have very simple licenses. They can be used in any kind of venture without consequence. In addition to this, these systems are often much more lightweight then their commercial counterparts as they are not designed for severely heavy duty assignments. There is also often a network of forums and other helpful resources that pertain to these engines – a feature which often characterizes software found in the open source community. The two best known examples that exhibit all these advantages and more are *Bullet* and *Box2D* [7].

CITATIONS

[1] C. Monks, P. Crossno, "Three Dimensional Visualization Of Proteins In Cellular Interactions," [Accessed 10/25/2014]

[2] M. Bose, V. Rajagopala, , "Physics Engine on Reconfigurable Processor - Low Power Optimized Solution empowering Next-Generation Graphics on Embedded Platforms," *The 17th International Conference on Computer Games*" [Accessed 10/26/2014]

[3] Z. Onyhak, K. Matkovic, H. Hauser, "Interactive 3D Visualization of Rigid Body Systems," *Visualization. IEEE*, Vol. , no. , pp. 539 - 546, [Accessed 10/25/2014]

[4] A. Kirsch, "Rigid Body Motion," Kairos [online]. http://blog.blackhc.net/wp-content/uploads/2010/05/Rigid-Body-Motion-Presentation.pdf. [Accessed 10/26/2014]

[5] R. Nave. "Description of Motion in Once Dimension," [online]. http://hyperphysics.phy-astr.gsu.edu/hbase/mot. [Accessed 10/26/2014]

[6] P. Pacheco, Introduction to Parallel Programming, 11d. ed. Morgan Kaufmann Publishers, 1993.

[7] "List of physics engines and reference material," [online]. http://www.gamedev.net/topic/475753-list-of-physics-engines-and-reference-material-updated-7-march-2011/. [Accessed 10/26/2014]

[8] "Havok Game Dynamics SDK, A Product Overview of the most comprehensive, optimized, real-time physics solution for game development." [Accessed 10/26/2014]