Efficient Rendering of Grass using Vertex Shaders

Taner Mansur

164820

Department of Computer Engineering,

Metu, 2009

Survey Papers

The papers I will analyze for this study will be generally related with grass rendering using vertex shaders.

Li's study [1] consists of an implementation of a Meadow (kind of a tree) simulation using vertex shader programming. In general they propose a simple to implement and efficient new technique on real-time animated grass rendering. They utilize animation with wind blow. In their implementation they model grass as 2 textured intersecting planes bound to a three segment stick lying in +y direction. With the effect of wind this stick shakes, its segments move to different locations in space and as a result textured planes are transformed on the scene. All the effects are achieved through vertex shader programs and their post-processing effects.

Bakay's paper [2] is about a simple method to render fields of grass, animated in the wind, in real time. Their technique use vertex shaders to render displacement of grass in the wind. Animation is achieved by translating the surface according to a local wind vector while preserving the length of the blades of grass. Their technique is fast enough to run efficiently on current common consumer graphics hardware and it can be also be applied to other similar surfaces such as hair and fur.

My Implementation

Analysis

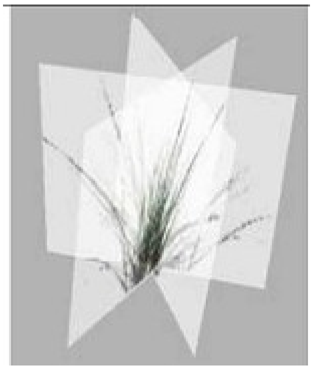
In this study I will make a survey on vertex shaders, their usage in CG, modern video games, current state of the art on shaders and their efficient implementation using Nvidia's Cuda [3] API. My survey will consist of papers on the current state of the art of vertex shading algorithms and their usage on rendering grass like objects. I will implement a grass rendering demonstration application that will use vertex shaders, texturing and lighting on OpenGL with CUDA.

A vertex shader is a graphics processing function used to add special effects to objects in a 3D environment by performing mathematical operations on the objects' vertex data. Each vertex can be defined by many different variables. For instance, a vertex is always defined by its location in a 3D environment using the x, y, z coordinates. Vertices may also be defined by colors, coordinates, textures and lighting characteristics. Vertex Shaders don't actually change the type of data; they simply change the values of the data, so that a vertex emerges with a different color, different textures, or a different position in space.

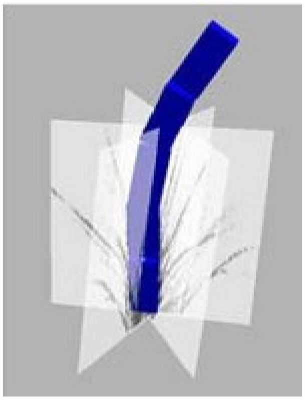
I will implement a modern looking grass rendering demo illustrating vertex shader programming and post-processing effects on GPUs. My hardware is my Dell laptop PC which

has an Intel Core2Duo T7300 2.0GHz Cpu, 3GB of RAM and a NVidia GeForce 8600GT-M

GPU. My GPU is capable of rendering vertex shader model v3.0 compliant shader programs. A grass object will be consisting of three textured planes intersecting each other through the middle axis (Img 1). And there will be another model which will be used in physical calculations (Img 2).



Img1 [1]. Grass object



Img2 [1]. Physical model

Physical model will bend in all directions according to the randomly generated wind force and using its current state textured planes will be rendered using vertex shading powered post-processing effects.

Implementation

CUDA is a general purpose parallel computing architecture that leverages the parallel compute engine in NVIDIA graphics processing units (GPUs) to solve many complex computational problems in a fraction of the time required on a CPU. It includes the CUDA Instruction Set Architecture (ISA) and the parallel compute engine in the GPU. To program to the CUDA architecture developers can use C which can then be run at great performance on a CUDA enabled processor.

Using OpenGL and CUDA an application will be developed to illustrate the use of vertex shading in real-time kinematics based animation. Physics will be computed rather on CPU or GPU. And results of physical calculations will then be applied to the rendered scene of a set of grass objects. The main aim is to keep all of the operations in real-time and to implement an efficient algorithm that will render a whole scene full of grass with an acceptable frame rate that will be around 25-30 FPS or more.

References

[1] 'Realtime Simulation of Meadow', C. Li, X. Guo, S. Lu, W. Wen, 2008 Asia Simulation Conference — 7th Intl. Conf. on Sys. Simulation and Scientific Computing

[2] B. Bakay, W. Heidrich, 'Realtime animated grass'. Proceedings of Eurographics 2002.

[3] L., P. and E. Schenk, "Shader-Driven Compilation of Rendering Assets", SIGGRAPH2002

[3] 'Programmable Pixel Shaders', NVidia Corp. 2009