Using 3D Graphics in Combination with other Models of Computation

Yasemin Demir, Man-Kit Leung

EECS 290N Report
May 15, 2009

University of California at Berkeley
Berkeley, CA, 94720, USA

Abstract

This paper presents a framework for developing a new graphics domain, called GRO, in Ptolemy II [1]. GRO is an OpenGL-based implementation that imports utilities, semantic and functionalities of OpenGL. The specification provides users to fully utilize the potential in the communication between the rendering engine and the computation in the model. This bottom-up design approach allows developers to create a event-based MoC that provides user, a sophisticated and exible specification. Based on OpenGL advantages, GRO is a user-friendly graphic domain that provides a declarative specification for the user to use. At the same time, it can also achieves better computation efficiency by migrating computation from the model (CPU) onto GPU's. Demos are provided showing the implementation of the new graphics domain.

1 Introduction

3D computer graphics is popular in modern technologies. 3D displays and real-time 3D viewing of modern systems, such as 3D human animations, tele-immersion systems, real-time organ viewing, entertainment environments and other applications of 3D computer graphics shows us the attractiveness of 3D technologies in nowadays life. Thus, 3D computer graphics turn out to be a significant tool that allows users and developers more entertaining and visual platforms and this off-course increases the share in the market.

Modern embedded systems also puts high demands on simulations using 3D graphics. This also increases the heterogeneity of embedded systems that is another important issue in modern technologies. However, this heterogeneity needs special environments that supports simulating each different subsystem and enabling the interaction between these subsystems. A real-time walking robot control system with a synchronous 3D visualization of its movements can be a good example for such heterogeneous embedded systems. In this example, a model of motion including calculations of next position and next movement can be a subsystem.

This makes Ptolemy II a good example for such environments that provides a broad range of computational models in an actor based platform. Ptolemy II allows users combine different models in different domains in the specified hierarchy. This advantage of Ptolemy II encourages developers to implement different platforms including such graphical interfaces, allowing the combination of 3D graphics with other models of computation (MoC) such as GR domain [2].

GR domain provides rendering of two-dimensional and three-dimensional graphics in Ptolemy II that is based on Java3D semantics and has a scheduling order based on scene graphs that are used to optimized the rendering of complex scene. It imposes certain constraints on scene objects. The representation is an directed acyclic graph which is a directed graph with no closed cycles. Each leaf node of a scene graph contain an object of within the scene to be drawn. The object can possibly be empty or null, in which case it is not rendered in the resulting image. Each immediate node (s.t. it has one or more children nodes) represents a coordinate space in which its children exist. The immediate node itself is a transformation that transforms the coordinate space. These transformations can composed together s.t. we can connect multiple transformation nodes before connecting to an object (leaf) node. There is one special node called the root, which is the original unchanged coordinate space-scene graphs are acyclic directed graphs where actors are connected in an acyclic directed graph. Based on scene graph semantics actors are fired according to GRScheduler that is done by performing a topological sort on all the ac-