Реализация модульного подхода при построении унифицированной системы научной визуализации

Антон Иванович Зенков
Уральский Государственный Университет
ИММ УрО РАН
Екатеринбург, Россия

Аннотация

В работе приводится описание разработанного подхода к унификации специализированной системы научной визуализации, описывается основанная на этом подходе программная реализация системы. Результаты данной работы основаны на достаточно большом опыте разработки систем визуализации.

Ключевые слова: когнитивная компьютерная графика, научная визуализация, Java3D .

1. ВВЕДЕНИЕ

В течение последних лет в ИММ УрО РАН были проведены исследования и разработки в области специализированных систем визуализации. Их результатом явилась реализация ряда специализированных систем визуализации, служащих в частности для представления данных, получаемых при численном решении задач оптимального управления и дифференциальных игр. В сотрудничестве с математиками был разработан набор средств, хорошо учитывающих специфику задач и ориентированных на выделение важных для исследователя особенностей исследуемых математических объектов. Однако при каждой новой постановке приходилось заново создавать не только те части системы, в которых реализуется специфика задачи и которые отвечают за восполнение/фильтрацию данных и мэппинг, но и непосредственно визуализационную часть системы, отвечающую за рендеринг, а также интерактивную оболочку, реализующую ставшие стандартными действия. Естественным решением этой проблемы является разработка унифицированной средств разработки специализированных систем визуализации.

Темой данной работы является обоснование данного подхода к разработке специализированных систем визуализации, а также описание конкретной программной реализации.

2. ТРЕБОВАНИЯ К УНИФИЦИРОВАННОЙ СИСТЕМЕ НАУЧНОЙ ВИЗУАЛИЗАЦИИ

Опыт работы с пользователями на этапе разработки специализированных систем визуализации показал, что пользователь хочет, может и должен принимать более активное участие в проектировании и реализации различных компонент системы, так как именно он знает, как должны выглядеть отображаемые модельные сущности. С другой стороны неправильно нагружать математика необходимостью разбираться в возможностях очередной графической среды и в тонкостях алгоритмов рендеринга. Пользователь-математик должен получить инструмент, позволяющий описать специфические для данной задачи процедуры восполнения/фильтрации и задать отображение (мэппинг) модельных сущностей на визуальные объекты, составляющие необходимый ему вид отображения.

Специализированные системы научной визуализации должны отвечать требованиям когнитивной (способствующей мышлению) визуализации [3,5], так как когнитивная визуализация обеспечивает эффективную интерпретацию результатов вычислений. Когнитивная визуализация должна осуществляться за счет разработки набора видов отображений. При этом полезным инструментом может выступать понятие метафоры. С ее помощью осуществляется перевод (переформулирование) понятий математической модели с языка строгих математических конструкций, где возможности нашей интуиции зачастую ограниченны, в визуальный. Язык визуализации можно рассматривать как набор графических образов, предъявляемых наблюдателю.

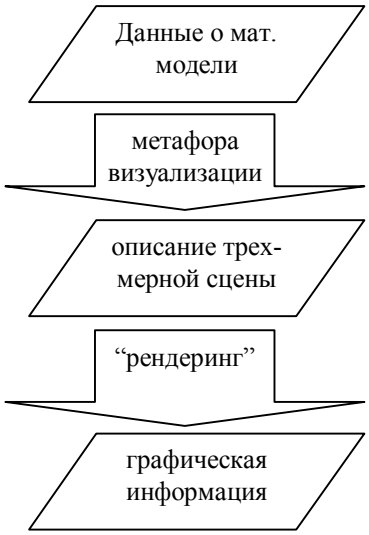

Рис.1 Схема визуализации

Словарь этого языка составляют виды отображения, используемые в той или иной системе [4]. На язык накладывается набор требований, которые в совокупности могут обеспечить когнитивную составляющую системы.

Декомпозиционный анализ функциональности созданных ранее систем привел к такой структуре унифицированной системы, в которой она содержит модуль визуализации, общий для различных специализированных систем, и набор модулей, восстанавливающих трехмерную сцену с учетом особенностей конкретных математических объектов. Для визуализации и исследования новых объектов пользователь-математик сможет разрабатывать собственные модули по заданной нами схеме. Формат входного потока данных такого

модуля согласован с соответствующей счетной программой, выходного - с модулем визуализации.

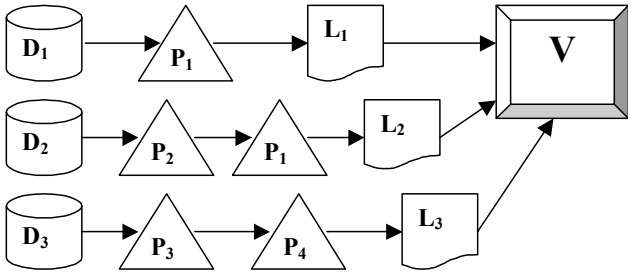

Рис.2 Схема работы системы

На входе имеются файлы с численными данными о неких математических объектах (D_1, D_2, D_3) разных типов. Для каждого типа файлов пользователем определяется последовательность модулей (P_1, P_2, P_3, P_4) в целях преобразования исходной информации в описание трехмерной сцены. Любая цепочка состоит из нескольких модулей (возможно одного), каждый из которых осуществляет некое преобразование данных. Первый модуль принимает на вход файл с исходной информацией, второй — результат работы первого и т.д. Формат промежуточных данных не оговаривается, и в каждой отдельной цепочке он может быть произвольным. Важно лишь, чтобы последний в цепочке модуль выдавал описание (L_1, L_2, L_3) сцены в рамках языка, о котором говорилось в первом разделе, в формате, принятом в модуле визуализации.

Согласно критериям выбора языка визуализации выбрана и описана конкретная реализация, используемая в нашей системе. Она предоставляет пользователю следующий набор трехмерных примитивов: точка, линия, треугольник. Каждый объект состоит из совокупности примитивов. Например, поверхности представляются как множество смежных треугольников с подобранными нормалями в вершинах (для передачи гладкости поверхности). Трехмерные тела представляются своей границей. Для каждого объекта пользователь независимо может задавать его визуальные атрибуты из достаточно богатого набора определенного заранее.

Для каждого типа входных данных должна быть создана цепочка программ-обработчиков, последний из которых должен на выходе создать набор данных установленного формата, содержащий 3-х мерную сцену.

3. ОПИСАНИЕ ПРОГРАММНОЙ РЕАЛИЗАЦИИ СИСТЕМЫ

Система представляет собой Java приложение. Использовался Sun JDK 1.3.1. Выбор Java был обусловлен хорошей поддержкой объектно-ориентированных технологий, богатым выбором свободно распространяемых вспомогательных библиотек (в частности разборщик языка XML — Xerces и библиотеки трехмерной графики — Java3D), а также возможностью работы на нескольких ОС.

Условно систему можно разделить на три основные части:

1. Чтение и обработка конфигурационного файла. Система запуска модулей обработки
2. Восстановление объектов, представляющих трехмерную сцену.
3. Интерфейс пользователя управления сценой и свойствами объектов.

Конфигурационный файл представляет собой данные в формате xml и состоит из трех разделов.

1. Описания всех обработчиков, для каждого: название, тип, строка запуска (для простых обработчиков), обрабатываемый тип файла.
2. Описание типов файлов, которые могут обрабатываться системой. Тип файла определяется по сигнатуре, расположенной в начале файла. Для каждого типа указывается последовательность вызовов обработчиков, которая необходима для получения правильной сцены.
3. Описание установок «по умолчанию» для разных свойств вновь загруженных объектов.

При запуске системы проверяется валидность конфигурационного файла и корректность цепочек обработчиков для всех типов файлов.

В системе реализовано два типа обработчиков: внешние исполняемые файлы и Java классы. Для того чтобы добиться корректной работы системы, обработчики в виде исполняемых файлов должны соблюдать определенный формат командной строки, а Java классы реализовывать специальный интерфейс.

После того как пользователь выбирает требуемый файл, система определяет его тип по сигнатуре и запускает соответствующую цепочку обработчиков, передавая, каждому следующему результат работы предыдущего в качестве входных данных. После того, как последний обработчик заканчивает работу, система должна получить файл с описанием трехмерной сцены в одном из поддерживаемых форматов.

Для разных задач оказываются удобными разные форматы. Для задач с относительно небольшими объемами вполне подходит формат текстовых файлов — XML, так как с его помощью можно легко описывать иерархические данные, и библиотеки разбора для этого формата общедоступны и хорошо отлажены. Однако для большинства задач этот формат не пригоден, так как числовые данные не оптимально с точки зрения объема представляются в текстовом формате. Естественно встает вопрос о двоичной форме представления сцены, в которой также может содержаться информация об иерархии и о свойствах объектов. Для поддержки нескольких форматов данных был создан универсальный класс загрузчика и его расширения для конкретных форматов.

Для вывода трехмерной графики и обеспечения интерактивной работы пользователя со сценой была использована высокоуровневая библиотека вывода трехмерной графики Java3D. Эта библиотека разработана фирмой Sun Microsystems и является свободно распространяемой. Java3D предоставляет программисту объектно-ориентированный набор высокоуровневых конструкций для рендеринга трехмерных сцен, а также структуры для целого ряда вспомогательных действий.

Система предоставляет пользователю 3 вида геометрических объектов:

1. Массив треугольников с нормалями в каждой точке.
2. Массив линий
3. Массив точек.

Каждый имеет набор визуальных свойств, как общих для всех объектов, так и индивидуальных. Например, общими свойствами являются — видимость, цвет, прозрачность, а индивидуальными — цвет изнанки для массива треугольников.

Несколько объектов могут быть объединены в группу. Группы объектов могут быть вложенными. Таким образом, мы

можем получить иерархическую структуру объектов. Для реализации такой функциональности был использован шаблон «Composite» [7].

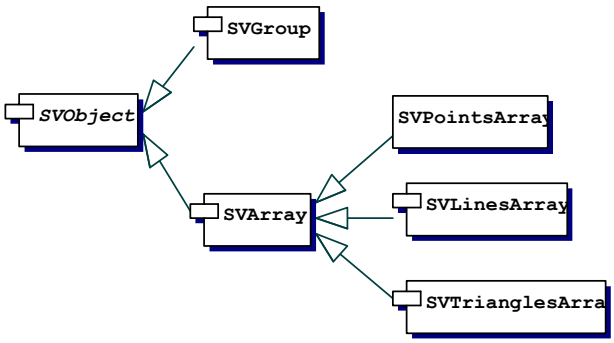

Рис.1 Диаграмма основных классов системы

SVObject представляет собой абстрактный класс, в котором определены методы для всех основных действий над объектами и их свойствами. От него наследуются класс, представляющий массивы графических примитивов (SVArray) и класс группы объектов (SVGroup). Такой подход позволяет унифицировать обращение к группе объектов и к конкретному объекту.

В данных классах также реализована связь системы с библиотекой Java3D.

На Рис. 4 изображено главное окно системы. Оно разделено на две части. Справа отображается сама сцена. Там же пользователь с помощью мыши может осуществлять всевозможные манипуляции с объектами сцены. Слева находятся элементы управления свойствами объектов и свойствами сцены.

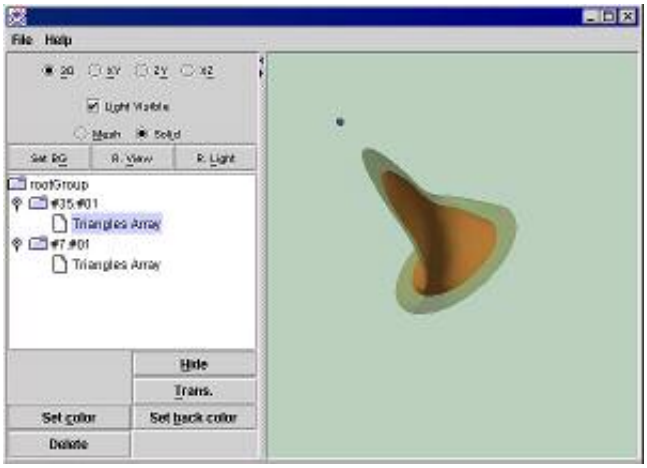

Рис.4 Пример интерфейса системы

4. ЗАКЛЮЧЕНИЕ

Разработанная система передана пользователям и находится на стадии достаточно интенсивной опытной эксплуатации. С помощью данной системы впервые произведена визуализация для ряда задач теории игр. Планируется развитие системы по двум направлениям.

1. Разработка модулей-обработчиков для различных математических объектов, то есть привлечение новых пользователей-математиков, а также разработка полезных и по возможности универсальных сервисных модулей, выполняющий типовые задачи компьютерной графики.
2. Дальнейшее наращивание функциональности модуля визуализации.

5. СПИСОК ЛИТЕРАТУРЫ

[1] Авербух В.Л., Зенков А.И., Исмагилов Т.Р., Манаков Д.В., Пыхтеев О.А., Юртаев Д.А. *Алгоритмы и программные средства параллельных вычислений. Выпуск. 4. Екатеринбург.2000. С. 3-23.*

[2] Авербух В.Л. *Метафоры визуализации // Программирование. 2001. No 5. С. 13-17.*

[3] Зенков А.И.*Разработка унифицированного модуля для специализированных систем научной визуализации. // Алгоритмы и программные средства параллельных вычислений. Выпуск. 5. Екатеринбург.2001. С. 3-23.*

[4] Zenkov A.I. *The specialized systems of scientific off-line visualization. // Труды 11 международной конференции по компьютерной графике и машинному зрению ``Графикон 2001", НГГУ. Нижний Новгород. 2001. С. 86-87.*

[5] Зенкин А.А. *Когнитивная компьютерная графика. М.: Наука. 1991.*

[6] Sun Microsystems, The Java3D API Specification, Version 1.2, April, 2000

[7] Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides *Design Patterns // Addison Wesley Longman, 1994*

Об авторе

Зенков Антон Иванович, аспирант кафедры информатики и процессов управления Уральского Государственного Университета, программист ИММ УрО РАН, г. Екатеринбург.

e-mail: **anton@internethome.ru**

Abstract

In this work the description of an attempt of unifying the specialized system of scientific visualization is given.

Работа выполнена при поддержке РФФИ, грант N 01-07-90210.