Министерство образования Российской Федерации
ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
Факультет информатики
Кафедра теоретических основ информатики

УДК 681.3.01/068

ДОПУСТИТЬ К ЗАЩИТЕ В ГАК
Зав. кафедрой, доцент, д.т.н.
_____ Ю.Л. Костюк
«___»_____ 2003 г.

Решетников Дмитрий Сергеевич

Разработка механизма конвертации геоданных для построения трехмерной модели местности с использованием VRML

Дипломная работа

Научный руководитель,
ассистент каф. ТОИ С.Г. Толузаков

Исполнитель,
студ. гр. 1481 Д.С. Решетников

Электронная версия дипломной работы помещена
в электронную библиотеку. Файл
Администратор

Томск 2003

РЕФЕРАТ

Дипломная работа 42 с., 13 рис., 8 источников, 2 приложения.

ТРЕХМЕРНОЕ МОДЕЛИРОВАНИЕ, ВИРТУАЛЬНЫЙ МИР, ВИРТУАЛЬНЫЙ ГОРОД, ВИРТУАЛЬНЫЕ МИРЫ В СЕТИ INTERNET, ЯЗЫК МОДЕЛИРОВАНИЯ ВИРТУАЛЬНОЙ РЕАЛЬНОСТИ VRML, VRML 97, КОНВЕРТАЦИЯ ДАННЫХ, ОБМЕННЫЙ ФОРМАТ MAPINFO INTERCHANGE

Объект исследования – технология создания виртуальных миров средствами языка VRML для построения трехмерных моделей и последующего их размещения в сети интернет.

Цель работы – разработка и реализация механизма генерации трехмерных объектов виртуальной реальности на основе геометрической и атрибутивной информации файлов обменного формата MapInfo Interchange с возможностями использования готовых объектов-примитивов, текстур и ссылок на внешние Web-объекты.

Методология проведения работы – изучение спецификаций языка VRML 97 и обменного формата MapInfo Interchange, создание инструмента генерации трехмерных объектов на основе геометрической и атрибутивной информации.

Результаты работы – разработан механизм генерации трехмерных объектов местности средствами VRML 97 на основе данных геоинформационных систем в формате MapInfo Interchange. Данный механизм реализован в виде оконного приложения Win32, работающего в диалоговом режиме.

Степень внедрения – разработанный программный продукт находится в опытной эксплуатации.

Прогноз о развитии и внедрения результатов исследования:

1) механизм генерации может быть использован в геоинформационных системах, требующих трехмерную визуализацию географических объектов и их публикацию в сетях Intranet/Internet;
2) система может быть расширена визуальным редактором наложения текстур для придания большей реалистичности отдельным элементам объектов VRML-сценам;
3) добавление JavaScript-объектов позволит создавать динамически изменяющиеся трехмерные модели местности.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Стремительное развитие информационных и телекоммуникационных технологий, в частности технологии всемирной сети World Wide Web, предполагает переход на более совершенный уровень изучения и обмена информацией. Статические текстовые блоки информации уступают место интерактивным анимациям, аудио/видеопотокам и средствам виртуальной реальности.

Уже сейчас многие разработчики геоинформационных систем дополняют свои продукты модулями работы с трехмерными векторными и растровыми объектами [7], позволяющими работать как со специальными 3D-структурами, так и с обычными двумерными картографическими проекциями, трехмерность которых восстанавливается по различным атрибутам. Разрабатываются и специальные средства публикации геоинформационных данных в локальных и глобальных сетях [8]. В качестве клиентских приложений таких систем выступают как самые простые, поддерживающиеся стандартными браузерами, динамические HTML-страницы и Java-апплеты, так и самостоятельные Java-приложения и отдельные модули геоинформационных систем, требующие предварительной инсталляции на компьютере пользователя.

В данной работе описываются основные проблемы и задачи трехмерного моделирования городских объектов в геоинформационных системах, различные технологии передачи и визуализации 3D-объектов в сети интернет. Отдельно рассматриваются возможности языка моделирования миров виртуальной реальности VRML – одного из лучших на сегодняшний день средства представления интерактивных трехмерных объектов в сетях интернет и интранет – для решения задачи построения трехмерной модели местности на основе плоских картографических данных, представленных в обменном формате MapInfo Interchange.

Разработанный механизм генерации трехмерных объектов виртуального мира на основе исходных данных формата MapInfo реализован в виде отдельного приложения и может быть использован для построения презентационных трехмерных моделей местности.

1. ТЕХНОЛОГИИ ТРЕХМЕРНОГО МОДЕЛИРОВАНИЯ

1.1. 3D-моделирование в геоинформационных системах. Технологии построения трехмерных моделей городских объектов

В настоящее время все большее количество геоинформационных систем стали дополняться средствами работы не только с плоскими, но и с пространственными геометрическими объектами. Трехмерные объекты в данных системах представляются либо в виде специальных векторных 3D-структур, либо в виде растровых поверхностей, содержащих информацию о третьей координате. В большинстве случаев трехмерные объекты удается восстановить из уже имеющихся двумерных. При этом значение высоты таких объектов берется, например, из значений высотных отметок.

Когда речь заходит о работе с данными в трехмерной перспективе, возникает мысль и об их трехмерной визуализации. Добавление третьей координаты делает возможным не только реалистичное отображение объектов нашего мира. Просмотр информации в трех измерениях дает новый взгляд на данные, позволяет выявить те закономерности, которые были скрыты при отображении этих данных на плоской карте, а построение трехмерных моделей обеспечивает возможность проведения дополнительного анализа.

Подобные геоинформационные системы позволяют решать задачи трехмерного моделирования и проектирования объектов, имеющих координатную привязку к поверхности земли. С помощью подобных систем можно, в частности, визуализировать архитектурный ландшафт, включающий здания, сооружения и рельеф местности. Благодаря этому можно избавиться от трудоемкого и дорогостоящего макетирования.

Моделирование миров виртуальной реальности позволяет представлять города или отдельные районы в качестве виртуального мира, по которому обозреватель может свободно перемещаться, ощущая все, что его окружает, так, как оно существует сегодня, существовало 10 лет назад, и так, как оно может выглядеть в будущем. Виртуальные миры позволяют “удалять” имеющиеся здания, заменив их парками или новыми архитектурными комплексами, запрашивать разнообразную информацию из баз данных об увиденных зданиях и местах.

Системы моделирования и визуализации виртуальных городов имеют огромное значение в процессе городского планирования и благоустройства [5]. Так, городские проектировщики при использовании подобных систем могут оценить физическое воздействие направляющих линий реализуемого проекта или сценариев землеиспользования при уточнении стратегических решений. Дизайнеры получают возможность мгновенной визуализации различных альтернатив для улучшения принимаемых решений. Городская администрация может лучше понять воздействие предлагаемого проекта на облик города. Проектировщики получают возможность проанализировать различные физические и финансовые аспекты проекта, и за счет этого повысить оптимальность принимаемых решений. Владельцы недвижимости и инвестор получают возможность визуализации влияния, которое проект окажет на сообщество, еще до начала его финансирования.

Значительных успехов в области виртуального моделирования добилась Группа Городского Моделирования (ГГМ) UCLA в проекте “Виртуальный город” [5]. С помощью разработанной уникальной системы и методологии компьютерного моделирования группа строит модели реального времени городских районов. Эти модели с точностью до надписей

на стенах и занавесок в окнах конструируются путем сочетания фотографий, полученных с помощью аэрофотосъемки, со стереометрическими представлениями улиц для реалистичного трехмерного моделирования. Разработанная группой методология построения виртуальных моделей подразделяется на три компонента:

- создание городской базы данных с помощью пакета трехмерного моделирования в реальном времени, данных о городе с привязкой к местным координатам и фотофактуры городских районов;
- интерактивная визуализация базы данных с использованием системы моделирования;
- сопровождение постоянно увеличивающейся базы данных с помощью сервера виртуального мира.

Процесс создания «городской базы данных» в методологии ГГМ разделяется на несколько этапов. Вначале происходит сбор информации из местных отделов городского планирования, которая включает в себя географические координаты, точные адреса, точные линеаризованные карты городских улиц. На эти карты накладываются аэрофотоснимки с отпечатками районов, парковых зон и других элементов ландшафта.

На следующем этапе моделирования собирается фотофактура фасада каждого здания каждой улицы. Собранные цифровые фотоснимки «накладываются» на трехмерные макеты зданий виртуального мира, полученные в результате компьютерного моделирования. На этом этап моделирования городской среды считается завершенным, и она готова для визуализации. В дальнейшем полученная модель дополняется информационным наполнением, необходимой для решения конкретных поставленных задач.

Виртуальные модели городских районов, построенные по методологии ГГМ, обладают высокой степенью реалистичности, но сам процесс моделирования требует большого количества исходных данных высокой точности. На данный момент существуют технологии, позволяющие получать трехмерные модели зданий и сооружений с гораздо меньшими затратами, например, технология создания трехмерных объектов по плоским проекциям, описанная в [6]. Основная идея этого метода заключается в восстановлении модели сооружения по его плоским проекциям, в качестве которых могут быть использованы чертежи, планы и любые другие доступные изображения. Система моделирования обрабатывает имеющиеся данные и в диалоговом режиме запрашивает у пользователя дополнительную информацию о геометрии объекта. В результате получается трехмерная модель здания, которая, при наличии среди исходных данных фотографий, может включать не только макет здания, но и текстуры его граней. Система позволяет строить сложные модели итеративно, начиная с модели с низким уровнем детализации, постепенно внося в нее более мелкие фрагменты.

В данной работе рассматривается механизм генерирования трехмерных объектов на основе имеющихся геометрических координат этих объектов на плоскости и дополнительных атрибутивных данных, представленных в обменном формате геоданных MapInfo Interchange.

1.2. Представление трехмерных объектов в Internet

Долгое время 3D-технологии активно применялись только в несетевых приложениях: кино, телевидении, играх, производственном проектировании и моделировании. Проникновение 3D-графики в Интернет сдерживалось из-за недостаточной скорости передачи данных и из-за отсутствия необходимых программных технологий адаптации 3D-файлов к Интернет-форматам. Во второй половине 90-х годов были созданы все предпосылки для интеграции 3D и Интернет технологий. Был разработан и стандартизирован язык моделирования виртуальной реальности VRML [1]. Появились программы, которые могли бы создавать и сохранять трехмерные сцены и модели в формате VRML [3]. Не меньше десяти фирм выпустили программы для просмотра VRML объектов, которые могли бы подключаться к работе основных браузеров в качестве plug-in. И, наконец, телекоммуникационные технологии шагнули так далеко, что сейчас речь идет о скоростях передачи информации по телефонным линиям до 1,5 Мбит/с, по кабелю - до 30 Мбит/с, по спутниковым каналам - до 40 Мбит/с.

Самым простым способом представления трехмерного изображения является *анимированный GIF*. При наличии достаточного количества графических изображений какого-то предмета, последовательно снятых с разных сторон, их можно свести в один GIF-файл, задав последовательность, скорость, фон отображения и некоторые другие параметры. Любой браузер отобразит этот файл в виде ролика. Существенными недостатками для практического применения данного средства являются значительный размер файла и отсутствие возможности взаимодействия с пользователем (нет возможности управлять процессом просмотра).

Формат IMOB (image objects) представляет собой коллекцию фотоизображений реального трехмерного объекта, сделанных из различных углов. Или наоборот, набор фотографий, сделанных из одной точки в разных направлениях, что в совокупности представляет панораму. При воспроизведении пользователь имеет возможность вращать, удалять и приближать объект. Таким образом достигается некоторая интерактивность в исследовании реального трехмерного объекта. Организовать просмотр IMOB-коллекций возможно на разных языках программирования: Java, JavaScript, ActiveX.

С помощью *технологии IPIX* (www.ipix.com) можно рассматривать объекты изнутри. Идея состоит в том, что создаются панорамные фотографии посредством цифровой камеры и даются возможности навигации в этой панораме.

Технология MetaStream позволяет дискретно приближать и отдалять объект, перемещать его на плоскости, выделять область изображения для того, чтобы рассмотреть ее крупным планом.

Продукты семейства Virtuoso призваны облегчить загрузку трехмерных интерактивных изображений в Web. Эту технологию оптимизации и отображения 3D объектов с 1997 года продвигает на рынок электронной коммерции израильская компания Virtue.

Технология cult3D похожа на предыдущую. Она также объединяет в себе plug-in для просмотра родных объектов на компьютере пользователя (в том числе и через Интернет) и программу для конвертации стандартной 3D-графики в свой формат. Результат визуализации примерно такой же, как и у Virtuoso.

Современная *технология производства визуальных эффектов Flash* компании MacroMedia постепенно завоевывает Интернет. Часто Flash применяют для динамических заставок, интерактивных рекламных роликов, организации многофункционального и разветвленного меню.

На языке *Java*, придавшем новый смысл интерактивным возможностям WWW, также конструируются эффектные 3D объекты. Java является интерпретируемым языком, недостаточно приспособленным для создания виртуальных миров. Вследствие этого апплеты Java долго загружаются на компьютер пользователя.

На базе Java и некоторых оригинальных технологий функционирует система *Shout3D*. Она позволяет включать в свои объекты цветовые эффекты, звук, прозрачный фон, морфинг и другие эффекты. Панель управления позволяет изменять некоторые настройки и навигацию по сцене (плавно и дискретно передвигать камеру обзора, приближать и отдалять модель).

3D-технологии от немецкой фирмы Dimension 3D-Systems, предлагают несколько программ, обеспечивающих трехмерное сканирование реальных предметов и отражение полученных 3D-моделей в окне браузера. При этом возможности просмотра ограничены только вращением модели вокруг различных осей.

3DML (3Dimensional Markup Language) – язык разметки третьего измерения. Создан в 1998 г. и ещё не стандартизирован. Он выполняет те же задачи, что и VRML (см. ниже), но не описывает сами объекты, а лишь строит из готовых блоков сцену. 3DML проигрывает VRML в гибкости и мощности, но выигрывает в простоте освоения и скорости передачи информации.

Самым лучшим на данный момент средством для построения трехмерных миров в Интернет является *VRML*, изначально разрабатываемый для использования в сетях Internet, Intranet и на локальных системах. Этот язык позволяет строить как статичные виртуальные миры, в которых обозреватель может перемещаться в произвольном направлении, с различной скоростью, так и использовать анимацию и скрипты для создания динамически изменяющихся миров. В 1998 году спецификация языка VRML 97 была утверждена организациями ISO и IEC в качестве международного стандарта.

X3D - следующее поколение спецификации VRML, разрабатываемой консорциумом Web3D. Как и VRML, этот язык описывает графические сцены в текстовом формате. Основное отличие заключается в том, что он использует формат XML вместо стиля OpenInventor, используемого в VRML 1.0/2.0/97.

2. VRML – ЯЗЫК МОДЕЛИРОВАНИЯ МИРОВ ВИРТУАЛЬНОЙ РЕАЛЬНОСТИ

2.1. Обзор языка VRML

Идея создания трехмерных виртуальных миров в World Wide Web зародилась несколько лет назад с появлением первой версии языка VRML (Virtual Reality Modeling Language) и с тех пор успешно развивались и пополнялись различными разработками. В настоящее время существует большое количество трехмерных миров, используемых на Web-узлах для маркетинга, продаж, дизайна, обучения и развлечений [2].

Язык VRML предназначен для описания трехмерных изображений и оперирует объектами, описывающими геометрические фигуры и их расположение в пространстве – фактически, полностью трехмерный виртуальный мир, по которому пользователь может перемещаться, манипулируя мышью, в любом направлении, с любой скоростью. Мир состоит из отдельных объектов, каждый из которых может «жить» своей жизнью, реагируя на внешние воздействия. И если спецификация VRML 1.0 предусматривала только статические миры, то благодаря последующим спецификациям (VRML 2.0, VRML 97 [1, 4]) появился целый ряд новых возможностей – таких как анимация и контроль над поведением объектов, музыкальное сопровождение, выявление столкновений (механизм collision detection), благодаря которому нельзя ходить сквозь «стены». Кроме того, имеется возможность учитывания физической природы и деформации объектов (например, можно создавать «желеобразные» структуры).

Формат VRML разрабатывался для использования в сетях Internet, Intranet и на локальных системах. Также предполагалось, что VRML будет универсальным обменным форматом для интегрированной 3D-графики и мультимедиа.

При разработке языка VRML использовались следующие критерии и требования:

- поддержка компьютерных программ для создания, редактирования и контроля VRML-файлов, а так же средств автоматического конвертирования из файлов широко используемых форматов трехмерных объектов в файлы VRML;
- обеспечение возможности использования и комбинирования динамических 3D-объектов в VRML-мирах и, как следствие, поддержка их повторного использования;
- обеспечение возможности добавления новых типов объектов, не определенных в VRML явно;
- поддержка реализуемости на широком спектре систем;
- исполняемость на большом количестве вычислительных платформ;
- возможность создания неограниченно больших динамических 3D-миров.

VRML имеет способность представлять статические и динамические 3D и мультимедийные объекты с гиперссылками на другие объекты, такие как текст, звуки, видеоклипы и изображения. VRML-браузеры, так же как и программы для создания файлов VRML, доступны для широкого спектра вычислительных платформ. VRML также поддерживает расширяемое моделирование, что позволяет определять и использовать новые динамические 3D-объекты.

Стандарт ISO/IEC 14772 – The Virtual Modeling Language [1], – определяет файловый формат, который интегрирует 3D-графику и мультимедиа. Концептуально, каждый VRML-файл является 3D-пространством, содержащим графические и аудио-объекты, которые могут динамически изменяться различными механизмами.

Семантика VRML описывает абстрактное функциональное поведение интерактивной 3D-мультимедийной информации. Стандарт ISO/IEC 14722 не определяет ни физические устройства, ни какие-либо иные реализационно-зависимые понятия (например, разрешение экрана и устройства ввода). Этот стандарт предназначается для большого многообразия устройств и приложений, и обеспечивает большую свободу действий в интерпретации и реализации функциональностей. Например, ISO/IEC 14722 не предполагает наличия мыши или 2D-монитора.

Каждый VRML-файл:

- однозначно устанавливает координатное пространство мира как для всех объектов, определенных в самом файле, так и для объектов, содержащихся в подключаемых VRML-файлах;
- явно определяет и формирует набор 3D- и мультимедиа-объектов;
- может определять гиперссылки на другие файлы и приложения;
- может определять поведение объектов.

Важной характеристикой VRML-файлов является возможность собирания нескольких файлов вместе посредством включения и связывания через гиперссылки. Так, например, файл *worldmap.wrl*, в котором определяется вид карты мира, может содержать ссылки на множество других VRML-представляющих собой различные города.

Иерархическое включение файлов позволяет создавать неограниченно большие, динамические миры. Тем не менее, VRML гарантирует, что каждый файл полностью определяется объектами, содержащимися в нем.

Другой немаловажной особенностью VRML является то, что он предназначен для использования в распределенном пространстве, таком как World Wide Web. В язык встроены различные объекты и механизмы для поддержки составных распределенных файлов:

- механизм встраивания (*inlining*) других VRML-файлов;
- гиперссылки на другие файлы;
- использование уже установившихся стандартами Internet и ISO других форматов файлов;
- определение стандартизованного синтаксиса.

2.2. Структура VRML-файла и его интерпретация браузерами

VRML-файл представляет собой обычный текстовый файл и состоит из следующих основных функциональных компонентов: заголовка, графа сцены (*scene graph*), прототипов и механизма маршрутизирования событий (*event routing*). Содержимое VRML-файла обрабатывается для представления и взаимодействия с пользователями программами-браузерами.

Как и в случае с HTML, один и тот же VRML-документ может выглядеть по-разному в разных VRML-браузерах. Кроме того, многие разработчики VRML-браузеров добавляют нестандартные расширения VRML в свой браузер.

Далее в этой главе будут расписаны составляющие VRML-файла

2.2.1. Заголовок VRML-файла

Как уже было сказано выше, VRML-файл представляет собой обычный текстовый файл. Для того чтобы VRML-браузер корректно распознал файл с VRML-кодом, в начале файла ставится специальный заголовок – *file header*. Для стандарта VRML 1.0 заголовок должен выглядеть следующим образом:

```
#VRML V1.0 ascii
```

Для файлов VRML 2.0 и VRML 97 заголовок должен иметь вид:

```
#VRML V2.0 utf8 [optional comment] <line terminator>
```

Заголовок обязательно должен находиться в первой строке файла, кроме того, перед знаком **#** не должно быть пробелов. Кроме этого между **#VRML** и **V2.0**, а так же между **V2.0** и **utf8** должен быть строго один символ пробела. Идентификатор **utf8** в заголовке файлов VRML 2.0 и VRML 97 указывает на то, что для отображения интернациональных символов в ISO/IEC 14722 используется кодировка UTF-8, определенная стандартом ISO/IEC 10646-1 (известном также как Unicode).

Следует отметить, что VRML код является регистрозависимым, то есть прописные и строчные буквы имеют различие. Все строки, начинающиеся значком #, кроме первой, считаются комментариями.

VRML-файл должен иметь расширение *.wrl*. Допускается использование свободно распространяемого архиватора *gzip* для сжатия исходного кода VRML-файла. В последнем случае, файл может иметь расширение *.wrz*, либо *.wrl.gz* для указания того, что файл был сжат.

2.2.2. Различия между VRML 1.0 и VRML 2.0/97

Миры, созданные в VRML 1.0 являются статическими. Миры VRML 2.0/97 уже могут двигаться, изменяться и взаимодействовать с посетителями этих миров.

Версия VRML 1.0 позволяла использовать следующие возможности:

- встроенные стандартные объекты-примитивы (Cube, Sphere, Cone, Cylinder, Text);
- построение произвольных объектов (surfaces, linesets, poinsets);
- возможность летать, ходить, исследовать сцены;
- использование источников освещения;
- установление предопределенных камер (точек наблюдения);
- наложение текстур на объекты;
- использование гиперссылок;
- определение и повторное использование объектов.

Версия VRML 2.0 включает в себя все функции VRML 1.0 плюс:

- возможность анимирования объектов;
- использование переключателей (triggers);
- использование сенсоров (sensors);
- применение скриптов (Java, JavaScript) для определения поведения объектов;
- построение объектов при помощи экструзии (extrusion);
- добавление фоновых цветов и текстур;
- звуковое сопровождение (WAV, MIDI);
- анимированные текстуры;
- маршрутизирование событий;
- определение и повторное использование объектов и поведений, а также эффективное добавление новых узлов к языку используя PROTO и EXTERNPROTO.

Другим немаловажным отличием является то, что VRML 97 является международной спецификацией, принятой Международной Организациям по Стандартам (International Organization for Standardization (ISO)) и Международной Электротехнической Комиссией (International Electrotechnical Commission (IEC)) в декабре 1997 (ISO/IEC-14772-1:1997), а VRML 1.0 – нет.

Многие браузеры VRML 97 читают и правильно визуализируют файлы VRML 1.0, но ни один VRML 1.0 браузер не читает и не показывает файлы VRML 97.

2.2.3. Единицы измерения и координатная система VRML-миров

ISO/IEC 14772 определяет в качестве единицы измерения виртуальных миров метр. Все остальные координатные системы могут быть построены посредством трансформации координатной системы виртуального мира. Углы измеряются в радианах. Все остальные значения выражаются как часть от единицы.

В качестве системы координат используется правосторонняя трехмерная декартова координатная система. Оси располагаются на экране следующим образом: X – горизонтально в плоскости экрана, Y – вертикально в плоскости экрана, Z – смотрит прямо на пользователя.

2.2.4. Граф сцены (scene graph) и механизм маршрутизирования событий (event routing) в VRML-мирах

VRML-файл содержит описание направленного ацикличного графа, содержащего один или несколько узлов-корней (*root nodes*), некоторые из которых могут содержать узлы-потомки (*children nodes*), составляя определённую иерархию узлов. Эта иерархия называется графом сцены. В задачу узлов входит описание объектов и их свойств. Иерархически сгруппированная геометрия графа служит для обеспечения аудиовизуального представления объектов виртуального мира, а так же содержит узлы, участвующие в генерации событий и их маршрутизации.

Для построения виртуальных миров VRML позволяет использовать встроенные в язык узлыобъекты-примитивы, а так же позволяет разработчикам создавать свои собственные объекты на основе уже имеющихся.

Иерархия трансформации (Transformation Hierarchy)

Иерархия трансформации включает в себя все те корневые узлы и их потомки, которые имеют конкретное место размещения в виртуальном мире. VRML также включает понятие *локальной координатной системы*, определяемой в терминах трансформации координатной системы узла-предка. Система координат, в которой отображаются узлы-корни, называется *системой координат виртуального мира*.

Задачей VRML-браузера является представление VRML-файла конечному пользователю. Это делается посредством представления именно иерархии трансформации, которая описывает прямо воспринимаемые части виртуального мира.

Иерархия трансформации графа сцены должна быть представима в виде направленного ацикличного графа. Результат становится неопределенным в случае, если какой-либо узел иерархии трансформации будет иметь себя в качестве собственного предка.

Маршрутизирование событий (Event Routing)

Некоторые VRML-узлы могут генерировать события в ответ на изменение их окружения, либо в процессе взаимодействия с пользователем. Маршрутизирование событий дает разработчикам VRML-миров механизм, отличный от иерархии графа сцены, через который возникающие события могут распространяться для произведения изменений в других узлах. Будучи единожды сгенерированными, события отправляются к узлам назначения по маршруту в соответствии с установленным временным порядком и обрабатываются получающими их узлами. Такая обработка может изменить состояние узла, вызвать генерацию дополнительных событий или даже изменить структуру графа сцены.

Узлы сценариев (*script nodes*) позволяют произвольную обработку событий, определенную разработчиком. Событие, полученное узлом сценариев, приводит к выполнению функции сценария, которая имеет возможность посылать события либо посредством обычного механизма маршрутизации событий, либо в обход этого механизма, напрямую к любому узлу, на который данный узел сценариев имеет ссылку. Кроме этого, сценарии могут динамически добавлять или удалять маршруты, тем самым изменяя топологию маршрутизации событий.

Идеальная модель событий обрабатывает все события немедленно, в порядке их генерации. Отметки времени служат двум целям. Во-первых, они являются концептуальным устройством, используемым для описания хронологического потока механизма событий. Это

устройство гарантирует, что детерминированный результат может быть достигнут реальной реализацией, которая ссылается на задержки обработки и на асинхронное взаимодействие с внешними устройствами. Во-вторых, временные отметки позволяют обрабатывать события в порядке их поступления в результате действий пользователя, либо по истечению определенного времени между событиями.

2.2.5. Представление VRML-миров и их взаимодействие с пользователем

Интерпретацию, выполнение и представление VRML-файлов обычно берет на себя механизм, известный как *браузер*. Само представление называется *виртуальным миром* (*virtual world*) и он навигируется в браузере человеком или механическим устройством, называемым *пользователем* (*user*). Виртуальный мир отображается в том виде, в котором он может наблюдаться с конкретного местоположения. Эта позиция и ориентация в пространстве виртуального мира называется *зрителем* (*viewer*). Браузер обеспечивает навигационные парадигмы (ходьба, полет и т.п.), которые позволяют пользователю перемещать камеру (зрителя) по виртуальному миру.

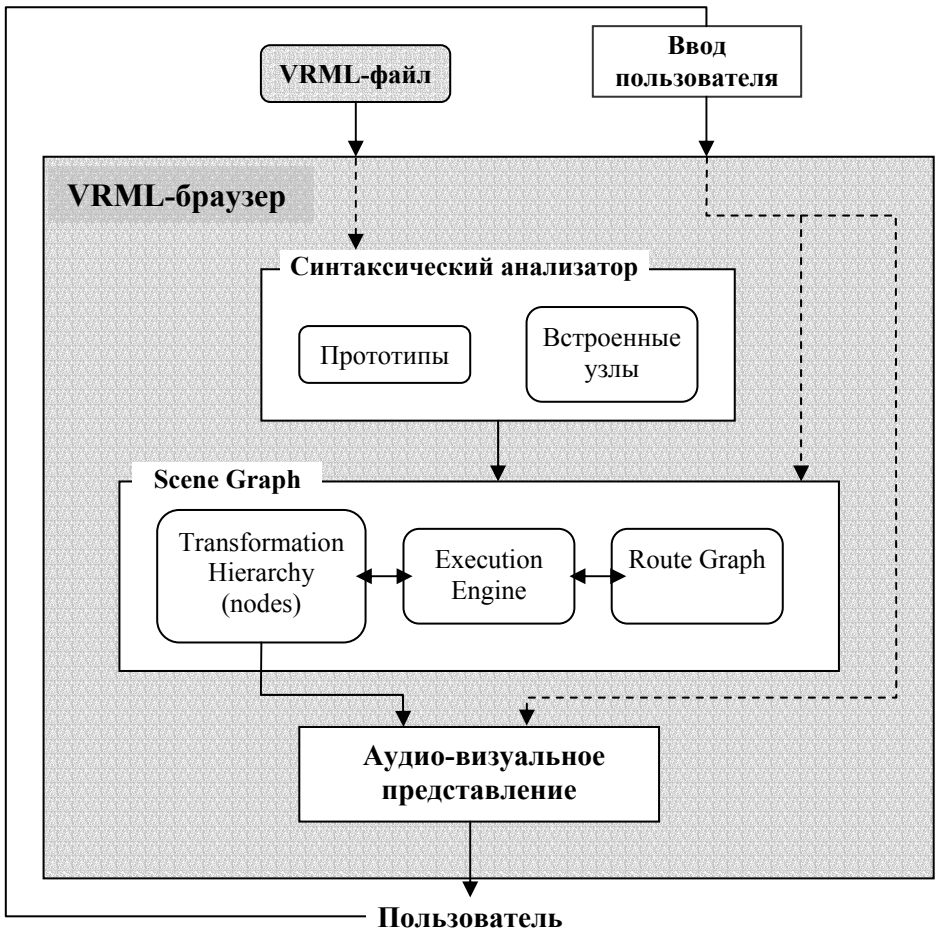

Рисунок 1. Концептуальная модель VRML-браузера

В добавление к навигации браузер обеспечивает механизм, позволяющий пользователю взаимодействовать с виртуальным миром через узлы-сенсоры (*sensor nodes*) в иерархии графа сцены. Сенсоры отвечают на взаимодействия пользователя с геометрическими объектами виртуального мира, передвижения пользователя по виртуальному миру или же на истечение интервалов времени.

Визуальное представление геометрических объектов в VRML-мирах следует концептуальной модели, предназначенной для получения сходства с физическими характеристиками света. Осветительная модель VRML (*lighting model*) описывает, как будут комбинироваться свойства внешнего представления объектов и источников освещения для получения отображаемых цветов.

Следует отметить, что спецификация языка VRML гарантирует отображение только внешней поверхности геометрических объектов. Вид «изнутри» спецификацией не оговаривается, и зависит от браузера. В большинстве случаев этого вида поверхностей нет, и браузеры отображают сцену так, как будто объекта, внутри которого находится текущая точка обзора, не существует.

На рисунке 1 приведена концептуальная модель VRML-браузера. Браузер представлен как приложение для отображения виртуальных миров, которое принимает пользовательский ввод (явный и косвенный) и передвижения пользователя (то есть манипуляции и навигацию посредством устройств ввода).

Тремя основными компонентами браузера являются: Синтаксический Анализатор VRML-кода (*Parser*), Граф Сцены (*Scene Graph*) и Аудио-визуальное Представление (*Audio/Visual Presentation*).

Синтаксический анализатор считывает VRML-файл и создает граф сцены. Граф сцены состоит из Иерархии Трансформации (*Transformation Hierarchy*), то есть непосредственно узлов, и Графа Маршрутов (*Route Graph*). Кроме того, граф сцены включает в себя Движок Исполнения (*Execution Engine*), который занимается обработкой событий, читает и редактирует граф маршрутов и вносит изменения в иерархию трансформации (в узлы). Пользовательский ввод обычно влияет на сенсоры и навигацию, и, таким образом, непосредственно связан с графом маршрутов (сенсоры) и компонентой аудио-визуального представления (навигация). Компонента аудио-визуального представления выполняет рендеринг графики и звука иерархии трансформации, которые отображаются пользователю.

2.3. Световая модель (lighting model) языка VRML

Световая модель (*lighting model*) языка VRML обеспечивает детализированные вычисления для определения цветов применительно к каждому геометрическому объекту сцены виртуального мира. Такие параметры, как цвет (задается значением узла *Color*), материал (*Material*), текстура (*ImageTexture*, *MovieTexture*, *PixelTexture*), уже примененные к конкретному объекту, объединяются с параметрами источников света, освещающих объект. Все эти вычисления были добавлены для имитации физических свойств светового потока, падающего на поверхности объектов

.

Для описания визуальных свойств объектов (материал поверхности, накладываемая текстура) используется узел *Appearance*:

```
Appearance {
  material              NULL
  texture               NULL
  textureTransform      NULL
}
```

Значение каждого из полей данного узла может быть *NULL*, в противном случае они должны содержать ровно один узел соответствующего типа. Так, поле *material* должно содержать узел *Material*. Если значение этого поля будет *NULL*, то при рендеринге соответствующих геометрических объектов будут проигнорированы все источники освещения VRML-сцены, а собственным цветом объектов будет белый (1, 1, 1).

Узел *Material* определяет свойства материала поверхности ассоциированных узлов геометрических объектов виртуального мира, которые используются при вычислении освещения при рендеринге:

```
Material {
  ambientIntensity      0.2
  diffuseColor          0.8 0.8 0.8
  emissiveColor         0 0 0
  shininess             0.2
  specularColor         0 0 0
  transparency          0
}
```

Значения всех полей узла *Material* могут варьироваться от 0.0 до 1.0 и определяют как будет отражаться свет от поверхности объекта для расчета цвета.

Раздел *ambientIntensity* указывает, как много света от окружающих источников света будет отражаться от поверхностей. Общее освещение (ambient light) является всенаправленным и зависит только от количества источников освещения, а не от их расположения относительно поверхности объекта. Общий цвет (ambient color) вычисляется как *ambientIntensity * diffuseColor*.

diffuseColor отражает все источники света сцены VRML в зависимости от угла наклона нормали поверхности объекта к источнику освещения. Поле *emissiveColor* позволяет моделировать “светящиеся” объекты.

Разделы *specularColor* и *shininess* определяют подсвеченные участки (например, светящееся пятно на яблоке). Когда угол от падающего на поверхность луча света становится близким к углу поверхности к обозревателю, *specularColor* к вычислениям рассеянного и общего цветов объекта. Меньшие значения поля *shininess* обеспечивают более мягкое свечение, в то время как большие значения параметра дают более четкое и меньше по площади подсвечивание. Поле *transparency* позволяет задать степень прозрачности объекта.

При необходимости на поверхность объектов виртуального мира можно натягивать текстуры различных форматов. Сам механизм наложения текстур состоит из двух частей, которыми занимаются соответствующие узлы. Узлы *ImageTexture*, *MovieTexture* и *PixelTexture* описывают что необходимо использовать в качестве текстуры, а вспомогательные узлы *TextureCoordinate* и *TextureTransform* задают как следует размещать текстуру на объекте.

Текстура назначается всему объекту в целом или какой-либо его части (грани). Есть три стандартных способа расположения текстуры на поверхности объекта (три системы координат, связанных с текстурой): плоский, цилиндрический и сферический. Кроме того, поведением текстуры можно управлять, изменяя ряд параметров (отражение, прозрачность, цветность, светимость и т.д.). Текстуры можно смешивать, используя маску прозрачности, определяющую, какая часть одной текстуры будет видна из-под другой.

В качестве текстур язык VRML позволяет использовать как статические растровые изображения, так и видеоролики. Спецификация предписывает браузерам поддерживать форматы JPEG и PNG, но обычно в этот список добавляется еще и GIF. Видеоизображения могут быть представлены в форматах MPEG-System (звук и изображение) и MPEG1-Video (только видеоизображение).

Shape-узлы сцены виртуального мира освещаются совокупностью всех источников света, под воздействие которых они попадают. Это могут быть как направленные, так и общие (рассеянные) источники освещения. Рассеянный свет связывается с отдельными источниками освещения сцены и является грубой аппроксимацией отражения световых потоков в реальном мире. VRML дает возможность использования следующих типов источников:

- *DirectionalLight* – направленный параллельный свет;
- *PointLight* – точечный всенаправленный источник;
- *SpotLight* – направленный расходящийся свет.

Все источники освещения содержат такие параметры, как яркость (*intensity*), цвет (*color*) и интенсивность рассеяния (*ambientIntensity*). Раздел *intensity* задает яркость прямого светоиспускания, а *ambientIntensity* определяет интенсивность рассеянного светоиспускания источника. Значение яркости может варьироваться от 0.0 (свечение отсутствует) до 1.0 (максимальная яркость). Поле *color* содержит информацию о цветовых свойствах светового потока в виде RGB-значения.

Узлы *PointLight* и *SpotLight* освещают все объекты виртуального мира, которые попадают под их воздействие независимо от местоположения этих объектов в иерархии трансформации. *DirectionalLight* освещает только те объекты, которые являются потомками узла, содержащего описание источника направленного параллельного освещения.

Узел *PointLight* служит для размещения в сцене точечного источника общего света, излучающего всем направлениям. Можно задать координаты источника (поле *location*), радиус сферы освещения (*radius*), а так же степень падения интенсивности по мере удаления от центра (*attenuation*):

```
PointLight {
  ambientIntensity        0
  attenuation             1 0 0
  color                   1 1 1
  intensity               1
  location                0 0 0
  on                      TRUE
  radius                  100
}
```

Три числа, указываемые для *attenuation*, используются в формуле для вычисления интенсивности на расстоянии *r* от центра:

$$I_r = \frac{I_0}{\max\left(1, attenuation_0 + attenuation_1 \cdot r + attenuation_2 \cdot r^2\right)}$$

Очевидно, что при *attenuation* 1 0 0 интенсивность меняться не будет.

Узел *DirectionalLight* задает освещение параллельными лучами в указанном направлении. По умолчанию это 0 0 -1, что означает направление точно от пользователя на экран. В связи с тем, что источник находится бесконечно далеко, не требуется задания его координат, а так же уровень затухания светового потока. Аналогом данного узла в реальном мире выступает Солнце:

```
DirectionalLight {
  ambientIntensity        0
  color                   1 1 1
  direction               0 0 -1
  intensity               1
  on                      TRUE
}
```

Параметр *intensity* задает яркость освещения, а *ambientIntensity* задает насколько велик вклад данного источника в общее освещение сцены за счет отражения и рассеяния от объектов.

Узел *SpotLight* является своего рода расширенной комбинацией двух предыдущих способов освещения: от *DirectionalLight* досталось *direction*, а от *PointLight* – *radius* и

attenuation. В результате получилось что-то вроде фонарика или прожектора: источник имеет положение и светит в определенном направлении:

```
SpotLight {
  ambientIntensity      0
  attenuation           1 0 0
  beamWidth             1.570796
  color                 1 1 1
  cutoffAngle           0.785398
  direction             0 0 -1
  intensity             1
  location              0 0 0
  on                    TRUE
  radius                100
}
```

Помимо затухания света при удалении от источника, которое регулируется параметром *attenuation*, данный узел позволяет также задать размывание по краям светового пятна. Для этого используются разделы *beamWidth* и *cutoffAngle*, означающие углы раствора двух конусов (см. рис. 2). Во внутреннем конусе (с углом раствора *beamWidth*) интенсивность в направлении перпендикулярно лучу постоянна и равна параметру *intensity*. Снаружи внешнего конуса (с углом *cutoffAngle*) интенсивность равна 0, а в зазоре между ними интенсивность спадает линейно. По умолчанию задается *beamWidth* > *cutoffAngle*, что дает световое пятно с неразмытыми краями.

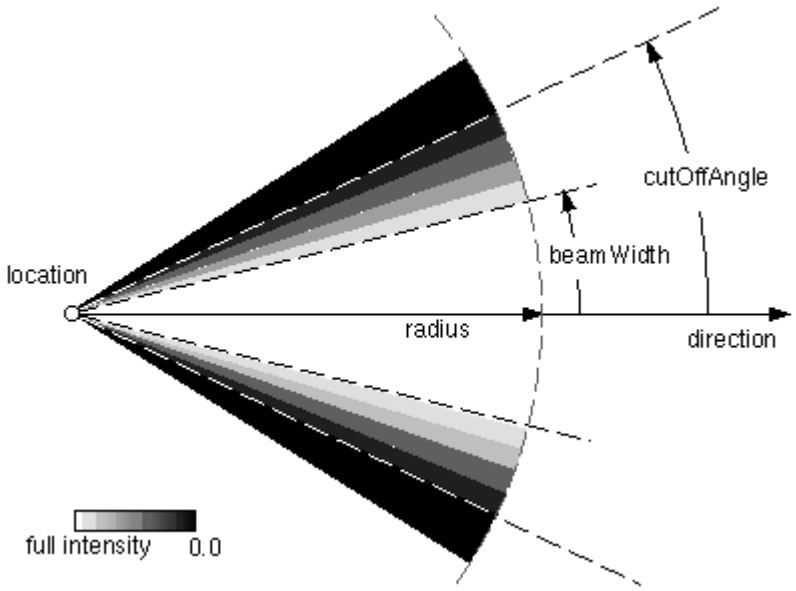

Рисунок 2. Узел SpotLight

Следует отметить, что модель освещения VRML не просчитывает тень, падающую от освещаемых объектов сцены.

2.4. Прототипы и повторное использование VRML-объектов

VRML предоставляет прекрасную возможность сократить и сделать более понятным исходный код VRML-файла посредством описания собственных объектов. Это значит, что если в изображении несколько раз повторяется одна и та же фигура, то ее можно описать всего лишь один раз и в дальнейшем только ссылаться на нее. Самым простым таким средством является использование директив языка *DEF*/*USE*.

При помощи ключевого слова *DEF* можно присвоить описываемому узлу некоторое имя, по которому в дальнейшем можно будет ссылаться на описание объекта в оператор *USE*. Директива *USE* не создает копий узла, а всего лишь повторно помещает один и тот же узел в граф сцены, в результате чего узел получает ещё одного родителя.

Действие имени узла ограничено одним VRML-файлом или разделом описания прототипа. В случае, если разным узлам присваивается одно и то же имя, оператор *USE* будет использовать ближайшую предшествующую ему вершину с заданным именем.

Ограничение области действия имен для повторного использования объектов можно обойти с помощью узла *Inline*:

```
Inline {
  url                   []
  bboxCenter            0 0 0
  bboxSize              -1 -1 -1
}
```

Данный узел является группирующим, и читает свою структуру (*children data*) из указанного в поле *url* местоположения в WWW. Время чтения структуры не определено, и некоторые браузеры во время загрузки данных отображают ограничивающий параллепипед узла. В качестве URL обязательно должен быть указан корректный VRML-файл.

Ограничивающий параллепипед (bounding box) служит оптимизационным целям при создании (размещении) и отображении загружаемых объектов сцены. По умолчанию его центр тяжести помещается в начало координат (определяется полем *bboxCenter*), а его размеры неограниченны (значение *bboxSize* = -1,-1,-1).

Прототипы позволяют проектировщику виртуальных миров расширять набор типов VRML-узлов. Определение прототипов может быть включено в файл, в котором они используются, либо могут быть определены внешне. Прототипы могут быть определены как в терминах других VRML-узлов, так и с использованием различных механизмов, специфичных для конкретного браузера. Хотя ISO/IEC 14772 содержит стандарт для определения подобных расширений, их реализация зависит от конкретного браузера.

Оператор *PROTO* позволяет определять новые типы узлов, так как если бы они являлись встроенными узлами VRML. Будучи единожды определенными, такие узлы могут использоваться в любом месте графа сцены так же, как и узлы предопределенных типов. Имена типов узлов должны быть уникальны для каждого VRML-файла. Результат несоблюдения уникальности именования спецификацией не определен.

При определении нового прототипа необходимо указать имя создаваемого типа объектов, а так же декларативный и определяющий разделы:

```
PROTO <name> [ <declaration> ] { <definition> }
```

Декларативная часть прототипов определяет поля, а так же входящие и исходящие события, обрабатываемые и генерируемые объектами нового типа. Описание включает в себя имена, типы и значения, присваиваемые по умолчанию каждому из полей или событий. Каждый экземпляр прототипа подразумевает собой полную копию прототипа со своими собственными значениями полей.

В определяющем разделе прототипа указывается один или несколько узлов, а также вложенные прототипы. Тип первого из узлов будет определять, как (в качестве узла какого типа) может быть использован данный прототип в VRML-файле. Так, если первый узел имеет тип *Material*, то такой прототип может быть использован везде, где требуется описание материала поверхностей объектов.

Экземпляры прототипов могут быть использованы в любом месте VRML-файла после полного описания прототипа. Прототипы не могут создавать собственные экземпляры в процессе своего описания.

При необходимости использования внешне определенных прототипов, можно воспользоваться оператором *EXTERNPROTO*, который позволяет загружать описания прототипов из других VRML-файлов, содержащих разделы *PROTO*:

```
EXTERNPROTO <name> [ <external declaration> ] URL or [ URLs ]
```

Применение узлов *Inline* и операторов *EXTERNPROTO* позволяет использовать распределенные в WWW библиотеки типов объектов виртуальных миров.

3. ИСХОДНАЯ ИНФОРМАЦИЯ ДЛЯ ПОСТРОЕНИЯ ТРЕХМЕРНОЙ МОДЕЛИ И МЕХАНИЗМ ЕЕ КОНВЕРТАЦИИ В VRML-КОД

3.1. Общая схема процесса построения трехмерной модели местности

Исходные данные для построения трехмерной модели местности представлены в файлах обменного формата MapInfo Interchange. Этот универсальный формат поддерживается большинством существующих геоинформационных систем и позволяет представить в виде обычных данных в виде ASCII-текста широкий спектр графических объектов. Благодаря особенности формата файлы MapInfo Interchange могут быть просмотрены и, при необходимости, откорректированы в любом текстовом редакторе.

Исходные данные, подготовленные в геоинформационной системе и выгруженные в файлы обменного формата MapInfo Interchange, поступают на вход механизма конвертации, который производит их обработку и осуществляет процесс формирования VRML-кода , представляющего собой трехмерную модель, полученную из представленных на входе двумерных данных с учетом атрибутики.

На рисунке 3 приведена общая схема процесса построения трехмерной модели местности:

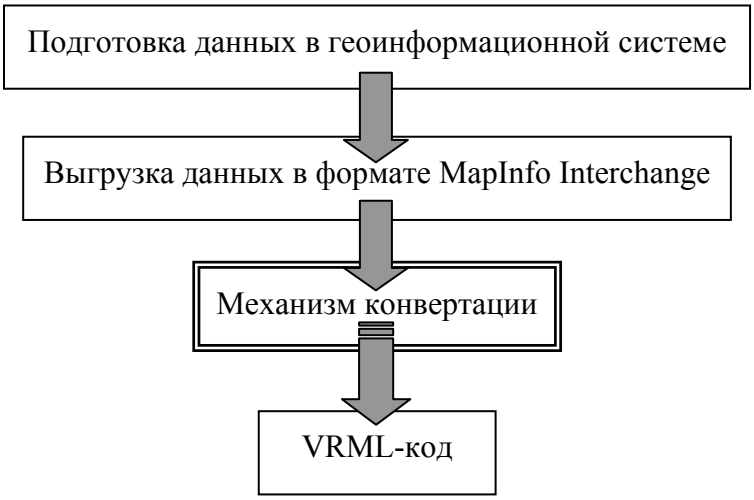

Рисунок 3. Процесс построения трехмерной модели местности

Поскольку исходные геоданные могут быть представлены в нескольких парах .MIF/.MID-файлов, которые должны быть сконвертированы в единый VRML-файл, приложение, реализующее механизм конвертации, должно сформировать общий заголовок VRML-файла, формат которого приведен в п. 2.2.1, и только после этого производить обработку отдельных пар файлов геоданных.

Далее механизм конвертации данных геоинформационной системы будет описан более подробно.

3.2. Структура исходной информации и анализ заголовочной части .MIF-файла

Геоданные, представленные в формате MapInfo Interchange, хранятся в парах файлов. Графическая информация размещается в .MIF-файлах, а атрибуты, связанные с соответствующими геометрическими объектами – в .MID-файлах. Информационные поля файла атрибутов отделяются разделителями, на каждую строку приходится одна запись. .MIF-файл состоит из двух разделов – заголовка и области данных. Заголовок содержит информацию об атрибутивных таблицах, а в области данных размещаются описания графических объектов (рис. 4).

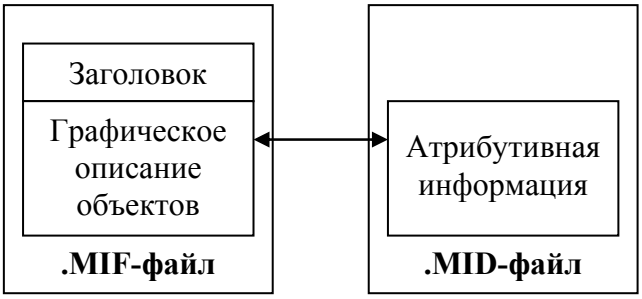

Рисунок 4. Структура файлов MapInfo Interchange

Механизм конвертации при разборе отдельной пары .MIF/.MID-файлов сначала должен произвести обработку информации полей заголовочной части .MIF-файла для корректного преобразования последующих данных. Общая схема работы механизма приведена на рисунке 5.

Заголовочная область .MIF-файла служит для задания параметров обмена геоданными и описания полей таблицы атрибутивных данных, связанных с геометри-ческими объектами. Эта часть имеет четко определенную структуру. Некоторые из разделов являются необязательными (указаны в квадратных скобках):

```
Version n
Charset "characterSetName"
[ DELIMITER "<c>" ]
[ UNIQUE n,n.. ]
[ INDEX n,n.. ]
[ COORDSYS... ]
[ TRANSFORM... ]
COLUMNS n
  <name> <type>
  <name> <type>
  .
  .
DATA
```

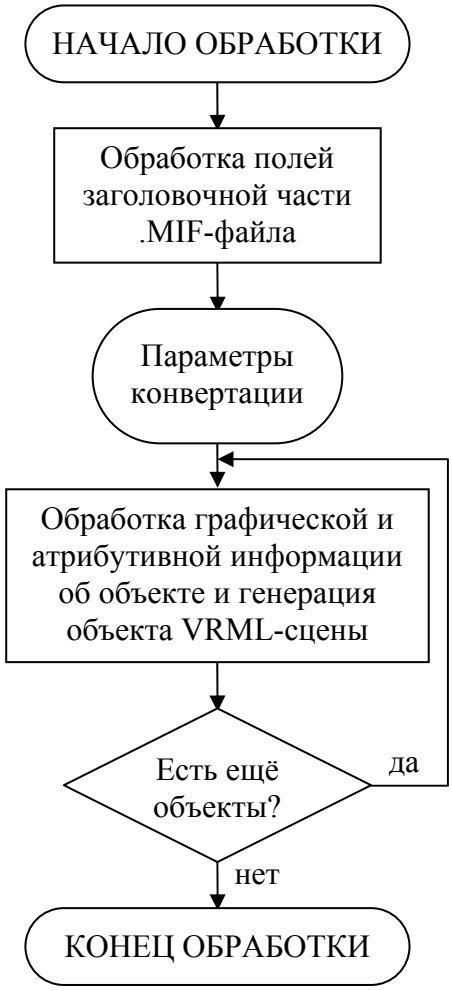

Рисунок 5. Механизм конвертации геоданных

Наиболее существенными для конвертации данных из файлов обменного формата в объекты виртуального мира являются поля *Version*, *Delimiter*, *CoordSys*, *Transform* и *Columns*. Значения остальных полей при конвертации могут быть опущены.

Параметр *Version* содержит информацию о версии формата (1, 2 или 300). Версия 2 дает возможности указания координатной системы (раздел *CoordSys*), в которой представлены геоданные и параметры их преобразования (*Transform*). Версия 300, реализованная в MapInfo 3.0, допускает описание полиполилиний, состоящих из нескольких участков.

Раздел *Delimiter* указывает на символ, являющимся разделителем полей данных в .MID-файле. По умолчанию таковым является символ табуляции.

CoordSys определяет, в каких координатах представлены графические данные в .MIF-файле, – в геодезических или в декартовых. Если данный раздел отсутствует в заголовочной части .MIF-файла, по умолчанию считается, что данные представлены в формате широта/долгота. Поскольку в VRML-мирах принята метрическая система декартовых координат на плоскости, механизм конвертации должен проверить наличие в заголовочной части .MIF-файла раздела *CoordSys* в следующем формате:

```
CoordSys Nonearth
  Units unitname
  Bounds (minx,miny) (maxx,maxy)
```

Поле *Units* определяет единицы измерения, а в *Bounds* задаются координаты прямоугольника, ограничивающего все объекты, описываемые в текущем .MIF-файле.

По умолчанию, все координаты в файлах обменного формата MapInfo приводятся относительно северо-восточного квадранта. Координаты объектов на территории США имеют отрицательную координату X, а координаты объектов в России и Европе (к востоку от Гринвича) имеют положительные координаты по оси X. Объектам в северном полушарии соответствуют положительные значения координат Y, а объектам из южного полушария – отрицательные. Раздел *Transform* используется для изменения координат точки отсчета и направления координатных осей:

```
TRANSFORM Xmultiplier, Ymultiplier, Xdisplacement,
  Ydisplacement
```

Значения полей данного раздела необходимо учитывать при обработке графического описания объектов области данных .MIF-файла и генерации соответствующих этим объектам элементов VRML-сцены.

Раздел *Columns* указывает количество атрибутивных полей записей, связанных с графическими объектами, описанными в разделе данных, а так же перечисление имен и типов этих полей:

```
COLUMNS n
  <name> <type>
```

```
    <name> <type>
    .
    .
DATA
```

В том случае, если механизм конвертации использует значения атрибутов из .MID-файла, выбирая их из фиксированных полей, он должен проверить наличие описания этих полей (их имен и типов) в заголовочной части .MIF-файла. В противном случае, обработку данного раздела можно опустить.

Обменный формат MapInfo Interchange допускает использование следующих типов данных:

- char – строковый, с указанием количества символов;
- integer – 4-байтовое целое со знаком;
- smallint – 2-байтовое целое со знаком;
- decimal – вещественное с фиксированным форматом целой и дробной части;
- float – вещественное;
- date – дата;
- logical – логическое значение.

При реализации механизма конвертации следует учитывать, что некоторые разделы заголовочной части могут быть опущены, и при необходимости использовать параметры, принимаемые форматом MapInfo Interchange по умолчанию.

3.3. Конвертация геообъектов в трехмерные объекты виртуального мира

Геометрическое и атрибутивное описание геообъектов разделяется на две части. Область данных .MIF-файла содержит информацию о графическом представлении объектов. Этот раздел начинается с ключевого слова DATA и может содержать произвольное количество описаний геометрических объектов. MapInfo ставит в соответствие записи в .MID- и .MIF-файлах, ассоциируя каждое описание графического объекта ровно с одной строкой атрибутов. Набор полей атрибутов является одинаковым для всех типов объектов и определяется в заголовочной части .MIF-файла (см. п. 3.2.). При необходимости введения атрибутов, не связанных с конкретным графическим объектом, имеется возможность создания “пустого” объекта, указав ключевое слово NONE в соответствующем месте .MIF-файла.

Допустимыми графическими объектами являются следующие типы:

- точка (point);
- линия (line);
- полилиния (polyline);
- регион (region);
- дуга (arc);
- текстовая надпись (text);

- прямоугольник (rectangle);
- скругленный прямоугольник (rounded rectangle);
- эллипс (ellipse).

Механизм конвертации при обработке отдельного объекта должен определить, к какому типу объектов он относится, вычленить необходимые значения атрибутивных полей, и вызвать определенную процедуру, производящую обработку объектов соответствующего типа (рис. 6).

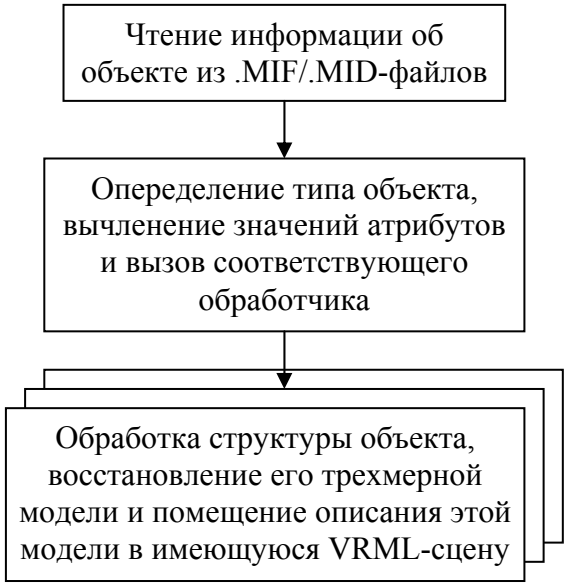

Рисунок 6. Обработка отдельного объекта

Далее будет рассмотрен механизм обработки объектов различных типов подробнее.

3.3.1. Обработка точечных объектов

Точечными структурами в геоинформационных системах определяются отдельно стоящие объекты реального мира (фонарные столбы, трубы, деревья и т.д.), которые при моделировании трехмерных элементов виртуального мира могут представляться соответствующими предопределенными обектами-примитивами из коллекции. Для объектов не имеющих определенной направленности (деревья, фонари, уличные урны) требуется указать только их координаты, а для таких объектов, как указатели, светофоры и т.п. необходимо дополнительно указывать угол, на который будут повернуты представляющие их трехмерные объекты VRML-сцены.

В файлах обменного формата MapInfo Interchange предусмотрено описание точечных структур в следующем виде:

```
POINT x y
  [ SYMBOL (shape, color, size) ]
```

Таким образом, графическое описание точечных объектов предоставляет информацию о координате расположения данного объекта на плоскости и, в некоторых случаях, символ, с помощью которого он отображается на плоской карте. Для механизма конвертации среди этих сведений являются полезными только координаты объекта на плоскости. Именно в этой точке следует размещать центр VRML-примитива, который будет представлять соответствующий точечный объект в трехмерной виртуальной сцене.

Среди атрибутов точечного объекта следует обязательно указывать имя файла, содержащего VRML-код соответствующего примитива. Данный файл будет подключен к VRML-сцене при помощи механизма «встраивания», описанного в п.2.4. Смещение вертикальной оси примитива осуществляется узлом *Transform*. Пример результатов преобразования показан на рисунке 7:

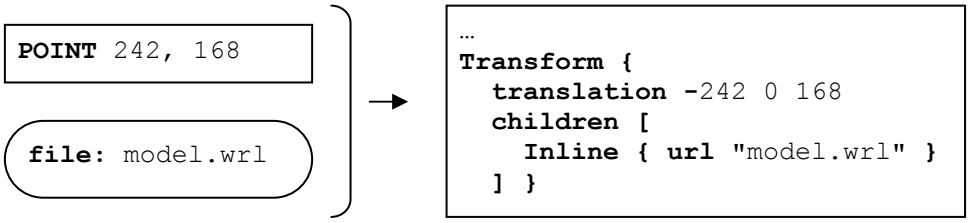

Рисунок 7. Пример преобразование точечных объектов

При необходимости, в атрибутах можно указать величину, на которую объект будет «приподнят» над поверхностью VRML-сцены. Значение такого атрибута должно быть указано в качестве второго параметра поля *translation* узла *Transform* (см. рис. 7).

Следует отметить, что поскольку в файлах MapInfo Interchange все координаты указаны относительно северо-восточного квадранта, при конвертации данных необходимо изменять направление оси X, изменяя знак соответствующей координаты. Кроме этого нужно учитывать параметры трансформации, приведенные в разделе *Transform* заголовочной части .MIF-файла.

Кроме визуализации отдельных точечных объектов плоской карты в некоторых случаях может потребоваться определение так называемых «точек обзора» (viewpoint) или «камер» VRML-сцены. Такие точки так же могут быть определены в геоинформационной системы в виде точечных объектов, у которых в качестве атрибута, указывающего на тип объекта следует указывать некоторое ключевое слово, которое будет сообщать механизму конвертации о необходимости размещения в точке с указанными координатами не визуального объекта, а камеры обзора. При этом кроме плоских координат требуется указание (в атрибутах точечного объекта) высотной координаты расположения точки обзора и, в случае необходимости, ее дополнительные параметры, такие как начальную ориентацию и угол обзора.

Поскольку такая информация является специфичной для конкретного типа элементов (в данном случае – точек обзора), добавление дополнительных полей атрибутов является нецелесообразным. Для подобных случаев схема данных может предусматривать наличие специального строкового поля, в котором пользователь сможет указывать необходимую дополнительную информацию, которая будет использоваться в неизменном виде при построении узлов VRML-сцены.

На рисунке 8 показан пример создания точки обзора на основе указанной графической и атрибутивной информации:

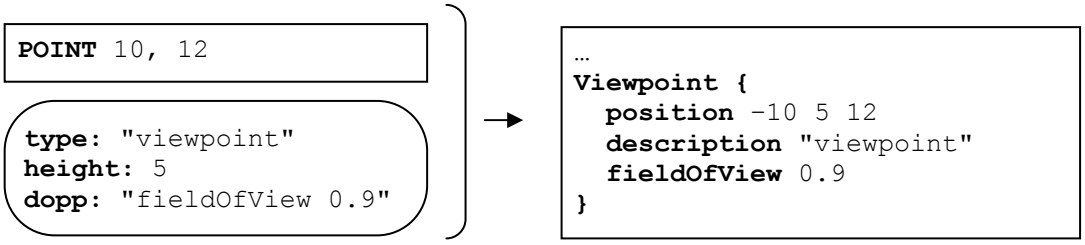

Рисунок 8. Добавление точки обзора VRML-сцены

3.3.2. Линейные и полилинейные объекты

Полилинии в геоинформационных системах служат для представления как действительно пространственных линейных объектов (таких как, например, линии электропереда), так и объектов, имеющих определенную трехмерную геометрическую структуру. И если в первом случае при конвертации полилиний геоинформационных данных в элементы виртуального мира достаточно использовать узел *IndexedLineSet*, который позволяет строить отрезки линий по набору координат точек, то во втором случае для моделирования следует применять либо экструзию (узел *Extrusion*) для представления таких объектов, как, например, бордюры, либо «собирать» цельные структуры из заранее подготовленных примитивов, следуя задаваемой полилинией траектории. В некоторых случаях полилинейные объекты требуется представлять в виде сочетания различных объектов-примитивов. Так, для моделирования ограждений, в координатах узлов полилинии следует помещать примитивы-столбики, а между этими узлами помещается некоторое количество элементов ограждений, зависящее от величины расстояния между «столбиками».

Поскольку описание трехмерного представления полилинейных объектов является в большинстве случаев проблематичным, рекомендуется преобразовывать эти объекты в наборы точек и полигонов, которые могут быть преобразованы соответствующими схемами конвертации в трехмерные объекты виртуального мира, описанными в данной работе.

3.3.3. Конвертация полигональных объектов

Полигональными объектами могут быть представлены контуры зданий, дорог, тротуаров и газонов. Как правило, данные контуры не имеют правильной геометрической формы и могут задаваться произвольным числом узловых точек, поэтому для построения их трехмерных эквивалентов целесообразно использовать такие структуры языка VRML, которые позволяют строить объекты виртуального мира по их координатам. Наиболее подходящим для создания подобных объектов является узел *Extrusion* (экструзия, выдавливание):

```
Extrusion {
  beginCap              TRUE
  ccw                   TRUE
  convex                TRUE
  creaseAngle           0
  crossSection          [1 1, 1 -1, -1 -1, -1 1, 1 1]
  endCap                TRUE
  orientation           0 0 1 0
  scale                 1 1
  solid                 TRUE
  spine                 [0 0 0, 0 1 0]
}
```

Данный узел позволяет создавать призматические трехмерные фигуры (а именно в таком виде и могут быть представлены полигональные геообъекты) избегая описания сложных трехмерных структур, состоящих из отдельных панелей-граней, как это делает узел *IndexedFaceSet*, очень часто используемый трехмерными редакторами.

Работает узел *Extrusion* следующим образом: сначала описывается многоугольник в плоскости с Y=0 (поэтому в разделе *crossSection*=”сечение” указываются только две координаты) и траектория его движения в пространстве. Поля *beginCap* и *endCap* определяют, будут ли создаваться грани-“крышки” на торцах результирующего объекта. В каждой точке траектории многоугольник можно:

- Масштабировать (раздел *scale*). Количество значений данного раздела должно быть либо 1 (тогда масштабируется исходное сечение *crossSection* и далее не изменяется), либо совпадать с количеством значений в траектории движения. Если количество значений в разделе *scale* больше единицы, но меньше количества значений в поле *spine*, то результат спецификацией VRML не определен. Благодаря масштабированию можно получать остроконечные объекты.
- Вращать (раздел *orientation*). Задается направление оси вращения (первые три числа) и угол в радианах. Вращение осуществляется по часовой стрелке относительно предыдущей точки траектории. Количество значений в разделе *orientation* определяется так же, как и для масштабирования.

В файлах MapInfo Interchange полигональные объекты могут быть представлены в виде регионов, которые, в свою очередь, могут состоять из одного или нескольких полигонов. Формат описания структуры-региона выглядит следующим образом:

```
REGION numpolygons
  numpts1
  x1 y1
  x2 y2
  . .
  [ numpts2
  x1 y1
  x2 y2 ]
  . .
    [ PEN (width, pattern, color) ]
    [ BRUSH (pattern, forecolor, backcolor) ]
    [ CENTER x y ]
```

Структура указывает общее количество полигонов в регионе (значение *numpolygons*) и для каждого полигона – количество его узлов (*numptsNNN*) с перечислением их координат. Дополнительно могут быть указаны свойства пера, кисти, а так же координаты центроида

Язык VRML не поддерживает полиполигональные структуры для описания трехмерных объектов виртуальной сцены, поэтому имеет смысл исходно разбивать подобные структуры на отдельные полигоны.

При построении трехмерной модели плоского полигонального геообъекта используется технология восстановления третьей координаты (высоты) из атрибутивных данных. Узлу *Extrusion* передаются координаты многоугольника сечения описываемого объекта плоскостью XZ (такая информация полностью берется из геометрических данных об объекте из .MIF-файла) и значение атрибута высоты. Траектория экструзии при этом будет состоять из одного участка, на протяжении которого не будут производиться ни масштабирование, ни вращение. На рисунке 9 показан пример преобразования структуры *Region* в узел *Extrusion* с использованием атрибута высоты:

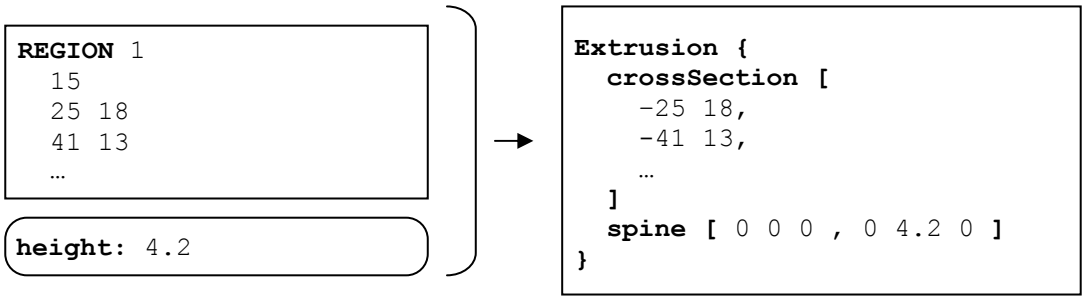

Рисунок 9. Конвертация объекта-региона

Узел *Extrusion* не является самостоятельным узлом VRML-сцены и не может быть указан в качестве одного из корневых узлов графа сцены. Его можно использовать как значение параметра *geometry* узла *Shape*:

```
Shape {
  appearance          NULL
  geometry            NULL
}
```

Этот узел является группирующим, и он позволяет указывать свойства материала поверхностей (см. п.2.3) всех объектов, описанных в составе этого узла. Свойства материала поверхности, а так же имя файла накладываемой текстуры могут быть частично получены из графических свойств объекта и указаны в качестве атрибутов.

Для придания большей реалистичности трехмерным объектам механизм конвертации может использовать узлы *Appearance* и *Material* (рис. 10). Цвет материала может быть использованный принятый по умолчанию языком VRML (белый), либо использовать цвет кисти (параметр *forecolor* раздела *BRUSH* структуры *Region*), используемый при раскраске обрабатываемого объекта геоинформационной системой. Следует учесть, что VRML для указания цвета использует три компоненты (RGB), значение каждой из которой может варьироваться от 0 (минимум яркости компоненты) до 1 (максимум). В файлах MapInfo цвета указываются в виде 32-битного целого, поэтому механизм конвертации должен осуществлять соответствующее преобразование.

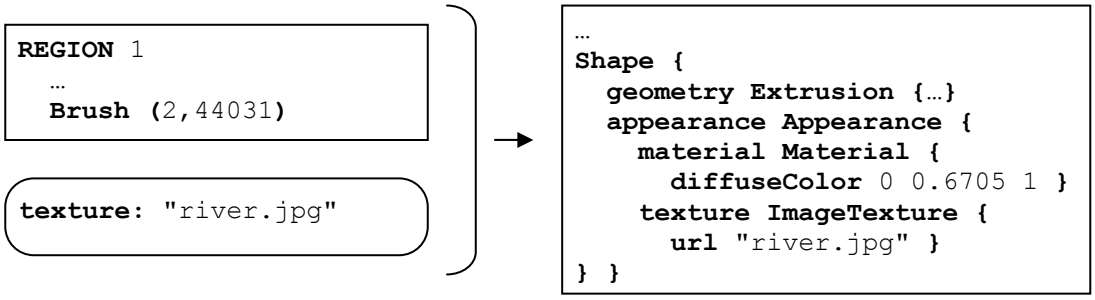

Рисунок 10. Указание цвета и текстуры поверхностей объектов

Текстуры следует применять для отображения структурных особенностей материала поверхностей, а так же для визуализации архитектурных деталей, которые по разным причинам не могут быть представлены в виде отдельных объектов. В качестве текстур могут быть использованы прямоугольные растровые изображения, полученные на основе имеющихся набросков и фотографий. Спецификация языка VRML предписывает браузерам

поддерживать форматы JPEG и PNG, но обычно допускается использование еще и GIF-текстур. Процесс подготовки текстур описывается в [6].

Для моделирования объектов со сложной пространственной геометрией рекомендуется производить их разбиение на более простые составные части. При этом может потребоваться указание высотной отметки, на которую будет поднят соответствующий элемент относительно поверхности «земли» (уровня с высотной координатой Y=0). «Поднятие» осуществляется применением узла *Transform* (аналогично его использованию для этих целей в п.3.3.1), в качестве значения поля *children* которого следует указывать полностью сформированный узел *Shape*.

3.4. Информационное наполнение трехмерной модели

Описанные выше схемы конвертации геоданных из формата MapInfo Interchange в трехмерные объекты VRML-мира осуществляют генерацию визуальных объектов графа сцены. В некоторых случаях имеет смысл осуществлять привязку геометрических объектов трехмерного мира к некоторой дополнительной информации.

VRML позволяет осуществлять переход на произвольные Web-объекты в окне браузера в ответ на клик мыши по соответствующим объектам виртуального мира. Это, в свою очередь, позволяет дополнять визуальную сцену произвольной информацией. Для установления ссылки на определенные URL служит невизуальный узел Anchor VRML-сцены:

```
Anchor {
  children        []
  description     ""
  parameter       []
  url             []
  bboxCenter      0 0 0
  bboxSize        -1 -1 -1
}
```

Данный узел является группирующим, и позволяет запрашивать содержимое указанного URL при активации (например, посредством клика мыши) пользователем любого из геометрических объектов, перечисленного в поле *children*. Если URL указывает на VRML-файл правильной структуры, этот мир заменит текущий мир в окне браузера. В случае, если ссылка устанавливается на объекты другого типа, браузер должен определить, каким образом следует их обработать. Как правило, такая ссылка передается в не VRML-браузер. В случае, если поле *url* указывается пустым, активация геометрических объектов не приводит к каким-либо действиям.

То, как происходит активация геометрических объектов и осуществляется переход на соответствующие объекты, зависит от типа используемого устройства ввода и определяется самим VRML-браузером.

Для реализации возможности перехода на произвольные Web-объекты при активации соответствующих визуальных объектов VRML-сцены, в исходных данных следует предусмотреть наличие атрибутивного поля, содержащего ссылку на требуемый ресурс, а механизм конвертации должен поддерживать обработку такого поля. При обработке объекта,

имеющего непустое поле URL, следует формировать узел *Anchor*, в поле *url* которого будет помещена ссылка на ресурс. При этом описание визуального представления трехмерногог объекта следует располагать не на уровне корневых узлов графа VRML-сцены, а помещать его в поле *children* сформированного узла *Anchor*.

ЗАКЛЮЧЕНИЕ

Идея создания трехмерных виртуальных миров в глобальной сети зародилась несколько лет назад с появлением первой версии языка VRML (Virtual Reality Modeling Language) и с тех пор успешно развивается и пополняется различными разработками. С помощью данной технологии имеется возможность представления в интернете как отдельных 3D-объектов, так и целых миров виртуальной реальности, описывающих целые города. Уже сейчас в сети существует большое количество трехмерных моделей, используемых на Web-узлах для маркетинга, продаж, дизайна, обучения, развлечений и демонстраций.

Простой текстовый формат VRML-файлов позволяет генерировать миры при помощи специальных приложений. Возможность повторного использования ранее описанных объектов облегчает процесс создания виртуальных миров и способствует значительному уменьшению объема передаваемой по сети информации.

В данной работе приводится описание схем разработанного автором механизма конвертации плоских геоданных в трехмерные объекты VRML-сцены. В качестве исходной информации для конвертации используются файлы широкораспространенного обменного формата MapInfo Interchange. Механизм реализован в виде отдельного приложения в среде Borland Delphi 5 и позволяет подключать внешнеопределенные VRML-структуры для визуализации отдельных объектов архитектуры, имеющих сложную геометрию, указывать свойства материала и текстуры поверхностей трехмерных элементов графа сцены и использовать механизм гиперссылок для информационного наполнения виртуальной модели местности.

СПИСОК ЛИТЕРАТУРЫ

1. ISO/IEC 14722-1:1997. Information technology. Computer graphics and image processing. The Virtual Reality Modelling Language (VRML). Part 1: Functional specification and UTF-8 encoding // VRML97 Specification, ISO/IEC 14772-1:1997 [Электронный ресурс]. – Режим доступа: http://www.vrml.org/Specifications/VRML97/index.html, свободный.
2. Федоров А. Трехмерные миры в World Wide Web // Компьютер Пресс, 1998, № 7. – С. 78-82.
3. Прыгин В. VRML 2.0: Moving Worlds средствами 3D-моделирования // Компьютер Пресс, 1998, № 5. – С. 78-88.
4. Авдеев М. Учебник по VRML 97 // [Электронный ресурс]. – Режим доступа: http://www.citforum.ru/internet/vrml97/index.shtml, свободный.
5. Красковский Д. Виртуальный город // Компьютер Пресс, 1998, № 7. – С. 112-116.
6. Костюк Ю.Л., Парамонов А.С., Гриценко В.Г. Технология создания трехмерных моделей объектов по плоским проекциям и ее применение в геоинформатике // Геоинформатика. Теория и практика. Выпуск 1. – Томск: Изд-во Томского университета, 1998. – С. 96–106.
7. Кищинская И., Лебедева Н. Дополнительные модули к настольным продуктам ArcGIS // ArcReview, 2001, № 4. – С. 9–11.
8. Калмыков Д. ArcIMS – Интернет ГИС нового поколения // ArcReview, 2001, № 4. – С. 12–13.

ПРИЛОЖЕНИЕ А. РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ

Требования к системе

Программа «VRML Converter» предназначена для работы на IBM-совместимых персональных компьютерах. Компьютер должен иметь:

- операционную систему Microsoft Windows 95/98/NT/2000/XP;
- процессор Intel Pentium 100 и выше;
- оперативную память 16 Мбайт и выше;
- VGA-совместимый дисплей

Программный продукт готов к работе и не требует предварительной установки. Однако для просмотра результатов конвертации может потребоваться установленный VRML-браузер, поддерживающий формат VRML 2.0.

Подготовка исходных данных

Исходные данные для программы «VRML Converter» должны быть представлены в файлах обменного формата MapInfo Interchange (.MIF/.MID), которые могут быть получены в результате экспорта данных из любой геоинформационной системы, поддерживающей данный формат. Данные должны быть представлены в плоской метрической системе декартовых координат. Дополнительные сведения о подготовке и выгрузке данных в файлы MapInfo Interchange можно получить в руководствах пользователя по имеющейся геоинформационной системе.

Программный продукт «VRML Converter» может обрабатывать векторные полигональные и точечные объекты геоинформационных систем. При этом полигональные объекты могут иметь в своей структуре до 256 узловых точек. Рекомендуется задавать полигоны ориентированными по часовой стрелке (в противном случае некоторые браузеры могут отображать такие полигоны некорректно). Полигоны, состоящие из нескольких областей не обрабатываются.

Географические данные картографических объектов должны быть снабжены атрибутивной информацией в виде следующих полей:

- *type* – char(…) – информация о типе объекта (здание, дорога, граница и т.п.). Для точечных объектов в данном поле требуется указать имя VRML-файла, который содержит визуальное представление данного объекта в виртуальном мире. В качестве таких VRML-объектов могут быть использованы как разработанные пользователем объекты, так и уже готовые модели сторонних разработчиков;
- *height* – float – высота объекта;
- *hdistance* – float – расстояние от поверхности земли до объекта;
- *texture* – char(…) – ссылка на файл текстуры, которая будет использоваться для придания трехмерному объекту реалистичности;
- *url* – char(…) – ссылка на Web-объект, по которому будет осуществляться переход при активации соответствующего объекта в окне VRML-браузера. Значение данного параметра требуется указывать при необходимости добавления дополнительной информации об объекте;

- *parameters* – char(…) – дополнительные параметры, которые будут учтены программой в процессе конвертации исходных данных об объекте. Здесь можно задавать параметры трансформации объекта (масштабирование, поворот и т.п.).

Работа с системой «VRML Converter»

Программный продукт «VRML Converter» выполнен в виде оконного приложения Windows и использует стандартные диалоги и элементы оконного интерфейса. Внешний вид программы представлен на рис.А.1:

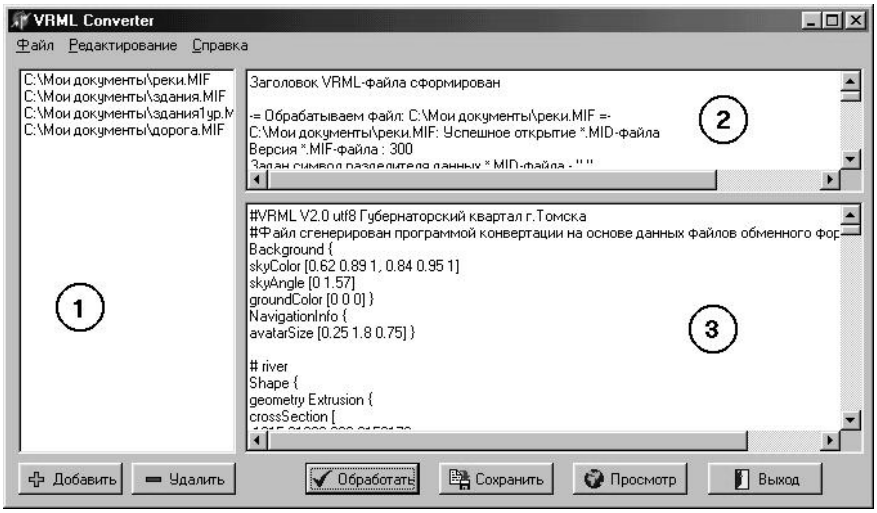

Рисунок А.1. Общий вид программы

Рабочее окно программы содержит строку меню, список файлов, подлежащий конвертации (1), историю процесса обработки (2), область просмотра и редактирования сгенерированного VRML-кода и строку кнопок.

Перед началом процесса генерации VRML-кода пользователь должен подготовить список обрабатываемых файлов формата MapInfo Interchange. Добавление файлов осуществляется нажатием кнопки «Добавить» или посредством выбора пункта меню «Файл\Открыть…». При этом открывается стандартное диалоговое окно открытия файлов, в котором пользователь может выбрать один или несколько *.MIF-файлов для последующей их обработки. Следует отметить, что при нажатии кнопки «Добавить» выбранные файлы добавляются к уже существующему списку, а при выборе пункта меню «Файл\Открыть» они заменяют файлы в списке.

Для удаления файлов из списка следует выбрать ненужные файлы и нажать кнопку «Удалить».

При изменении списка обрабатываемых файлов происходит очистка областей истории процесса конвертации и VRML-кода. Если текущий VRML-код не был сохранён, выдаётся соответствующее диалоговое окно, в котором пользователь может выбрать, следует ли сохранять текущий код.

После подготовки списка файлов, можно приступать непосредственно к генерации VRML-кода. Для запуска процесса генерации следует нажать кнопку «Обработать», либо

выбрать пункт меню «Редактирование\Конвертировать». Система начинает обрабатывать файлы из списка в порядке их перечисления. При этом в окне истории будут отображаться результаты выполнения процесса обработки элементов выбранных файлов, а в окне VRML-кода будут выводиться сгенерированные строки. По завершению процесса конвертации пользователю выдаётся сообщение о завершении обработки (рис. А.2).

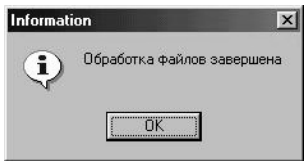

Рисунок А.2. Сообщение о завершении обработки

В ходе процесса обработки системой могут быть выданы сообщения об обнаруженных ошибках – об ошибках открытия .MIF/.MID-файлов, о неверном формате заголовочной части .MIF-файла, о несоответствии формата атрибутивных данных .MID-файла и т.д. При этом пользователю предоставляется возможность выбора вариантов дальнейших действий. В случае если ошибка связана с некорректностью атрибутивных данных, можно продолжить обработку со значениями атрибутов по умолчанию, прервать обработку некорректного файла, либо завершить процесс конвертации. Дополнительная информация о возникшей ошибке и о методе её решения может быть найдена в окне истории выполнения обработки. Пользователь может внести требуемые изменения в исходные данные и запустить процесс обработки повторно.

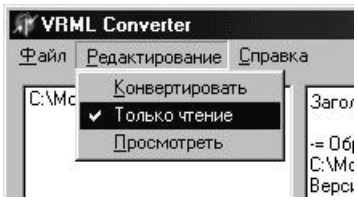

Рисунок А.3. Активация режима редактирования VRML-кода

Получившийся в результате генерации VRML-код отображается в текстовом виде в соответствующем окне (см. рис.А.1, поле 3). По умолчанию это поле доступно только для чтения, но если пользователю требуется внести коррективы, и он знаком со спецификацией языка VRML 97, он может активировать режим редактирования, сняв галочку с пункта меню «Редактирование\Только чтение» (рис. А.3).

Для сохранения сгенерированного и откорректированного VRML-кода следует нажать кнопку «Сохранить» или выбрать пункт меню «Файл\Сохранить». Пользователю будет выдан стандартный системный диалог сохранения файла. Файлы сохраняются с расширением .wrl. При выборе для сохранения имени уже существующего файла будет выдано соответствующее предупреждение.

При наличии установленного VRML-браузера, можно просмотреть результаты преобразования в виде трехмерных объектов виртуального мира. Нажатие кнопки «Просмотр» или выбор пункта меню «Редактирование\Просмотреть» передает имя файла, под которым были сохранены текущие результаты, зарегистрированному в системе средству просмотра VRML-файлов. Сведения о работе с конкретным просмотрщиком могут быть найдены в соответствующих руководствах пользователя.

Для завершения работы с программой служат стандартная кнопка закрытия окна, кнопка «Выход» и пункт меню «Файл\Выход».

ПРИЛОЖЕНИЕ Б. РУКОВОДСТВО ПРОГРАММИСТА

Общие характеристики проекта

Разработанный механизм генерации трехмерных объектов виртуального мира на основе геоинформационных данных, представленных в формате MapInfo Interchange, реализован в виде оконного приложения Win32 в среде Borland Delphi 5.1.

Проект приложения содержит один модуль и состоит из следующих файлов:

- ConverterPrj.dpr – файл проекта;
- ConverterPrj.dof – файл параметров проекта;
- ConverterPrj.cfg – файл конфигурации проекта;
- ConverterPrj.res – файл ресурсов проекта;
- Converter.pas – файл модуля проекта;
- Converter.dfm – файл описания формы модуля.

Требования к входным данным

Входные данные для программы «VRML Converter» должны представлять собой файлы обменного формата MapInfo Interchange (*.MIF/*.MID), содержащие геометрическую и атрибутивную информацию о картографических объектах. Геометрические данные должны быть представлены в декартовой системе координат. В качестве базовой единицы измерения должен быть указан метр.

Атрибутивная информация должна быть представлена в виде следующих полей (типы данных атрибутивных полей указаны в терминологии формата MapInfo Interchange):

- *type* – char(…) – здесь указывается информация о типе объекта (здание, дорога, граница и т.п.). Для точечных объектов в данном поле следует указывать имя VRML-файла, который содержит визуальное представление данного объекта в виртуальном мире. В качестве таких VRML-объектов могут быть использованы как разработанные пользователем объекты, так и уже готовые модели сторонних разработчиков;
- *height* – float – высота объекта;
- *hdistance* – float – расстояние от поверхности земли до объекта;
- *texture* – char(…) – ссылка на файл текстуры, которая будет использоваться для придания трехмерному объекту реалистичности;
- *url* – char(…) – ссылка на Web-объект, по которому будет осуществляться переход при активации соответствующего объекта в окне VRML-браузера. Значение данного параметра требуется указывать при необходимости добавления дополнительной информации об объекте;
- *parameters* – char(…) – дополнительные параметры, которые будут учтены программой в процессе конвертации исходных данных об объекте. Здесь можно задавать параметры трансформации объекта (масштабирование, поворот и т.п.).

Выходные данные

Результатом работы данной программы является VRML-код, содержащий информацию о трехмерных объектах виртуального мира, модели которых были получены в результате конвертации информации из входных файлов. При необходимости текст полученного VRML-кода может быть отредактирован в любом текстовом редакторе, либо прямо в соответствующем окне приложения.

Компоненты формы Convert.dfm

Ниже приводится список компонент, размещенных на форме модуля Converter и их назначение:

- `FileListBox: TListBox` – хранит список файлов, выбранных пользователем на обработку;
- `HistoryEdit: TRichEdit` – служит для отображения хода выполнения обработки файлов;
- `VRMLEdit: TRichEdit` – служит для хранения и отображения VRML-кода, генерируемого в процессе обработки файлов. Может быть доступным для внесения изменений;
- `AddBtn, DelBtn: TBitBtn` – кнопки, по нажатию которых производится добавление и удаление файлов в список FileListBox;
- `ProceedBtn: TBitBtn` – кнопка запуска процесса обработки списка файлов;
- `VRMLSaveBtn: TBitBtn` – кнопка вызова диалога сохранения VRML-кода;
- `WatchBtn: TBitBtn` – по нажатию этой кнопки происходит открытие предварительно сохраненного VRML-файла в VRML-браузере, зарегистрированном в системе;
- `CloseBtn: TBitBtn` – кнопка закрытия окна;
- `MainMenu1: TMainMenu` – строка главного меню;
- `MIFOpenDlg: TOpenDialog` – диалог выбора файлов для добавления в список. Допускает одновременный выбор нескольких файлов;
- `VRMLSaveDlg: TSaveDialog` – диалог сохранения VRML-кода.

Процедуры-обработчики

Процедура `FormCreate(Sender: TObject)` служит для заполнения переменных начальными параметрами во время запуска приложения.

Для подготовки списка файлов, для последующей обработки используются следующие процедуры:

- `procedure OpenBtnClick(Sender: TObject)` – вызывает диалоговое окно `MIFOpenDlg` для выбора файлов исходных данных. В случае подтвержения выбора файлов, производится очистка областей `HistoryEdit` и `VRMLEdit` и списка `FileListBox` с последующим его заполнением именами выбранных файлов. Если текущий VRML-код не был сохранен, выдается соответствующее предупреждение;

- `procedure AddBtnClick(Sender: TObject)` – вызывает диалоговое окно `MIFOpenDlg` для выбора файлов исходных данных, которые будут добавлены к текущему списку;
- `procedure DelBtnClick(Sender: TObject)` – удаляет выбранные имена файлов из списка.

Следующие процедуры осуществляют работу с полученном в результате работы системы VRML-кодом:

- `procedure VRMLSaveBtnClick(Sender: TObject)` – вызывает диалоговое окно `VRMLSaveDlg` для выбора имени сохраняемого VRML-файла и сохраняет содержимое окна `VRMLEdit` в файле с указанным именем;
- `procedure ReadOnlyBtnClick(Sender: TObject)` – управляет свойством ReadOnly окна `VRMLEdit`, содержащего VRML-код;
- `procedure WatchBtnClick(Sender: TObject)`- передает имя сохраненного VRML-файла функции `ShellExecute` для его запуска в VRML-браузере, зарегистрированном в системе.

Реализация механизма конвертации

Кнопка `ProceedBtn` и пункт меню `ProceedMnu` имеют в качестве обработчика процедуру `ProceedBtnClick(Sender: TObject)`, которая начинает процесс конвертации файлов из списка в трехмерные объекты виртуального мира.

Процедуры и функции, реализующие механизм конвертации геоданных в трехмерные объекты виртуального мира используют следующие глобальные переменные:

- `MIFFile: Text` – дескриптор текущего открытого .MIF-файла;
- `MIDFile: Text` – дескриптор текущего открытого .MID-файла;
- `IsMIDFileOpened: Boolean` – логическая переменная, указывающая на наличие открытого .MID-файла, кодержащего атрибутивную информацию о геометрических объектах, описание которых содержится в обрабатываемом .MIF-файле;
- `MIDDelimiter: Char` – символ-разделитель полей в файле атрибутов;
- `PointSet: array[1..256] of String` – массив строк, описывающих координаты узлов текущего обрабатываемого геометрического объекта.

Далее приводится описание процедур и функций, реализующих механизм обработки исходных файлов данных:

`procedure ProceedBtnClick(Sender: TObject)` – проверяет текущий VRML-кода на сохраненность и при необходимости предлагает пользователю сохранить результаты предыдущего преобразования, либо отказаться от запуска нового процесса обработки. После проверки на сохраненность вызывается процедура формирования заголовочной части будущего VRML-файла, а затем для каждой строки списка FileListBox вызывается функция обработки отдельной пары .MIF/.MID-файлов.

`procedure VRMLHeaderCreate` – помещает в окно VRML-кода фиксированный заголовок, наличие которого требует спецификация ISO/IEC 14772-1:1997.

`function ProceedMIFFile(MIFFileName: String): Boolean` – функция осуществляет предобработку .MIF-файла, имя которого передается ей в качестве параметра – открытие, поиск и открытие соответствующего .MID-файла с атрибутивной информацией, вызов функций проверки корректности загололовка .MIF-файла. В случае соответствия заголовка требованиям системы, производится вызов процедуры конвертации геообъектов в трехмерные объекты VRML-мира. По завершению обработки функция производит закрытие всех открытых файлов и добавляет сообщение об успешности завершения обработки текущей пары файлов. Функция возвращает логическое значение, которое имеет значение ложь если в процессе выполнения возникли ошибки ввода-вывода или .MIF-файл имеет неверный заголовок и пользователь выбрал вариант прекращения процесса дальнейшей конвертации.

`function MIFHeaderProceed: Boolean` – функция осуществляет проверку заголовка текущего открытого .MIF-файла на соответствие требованиям механизма конвертации. Выясняется версия .MIF-файла (используется только для информационных целей), устанавливается символ разделителя атрибутивных данных соответствующего .MID-файла, проверяется используемая система координат. В случае, если до вызова данной функции был успешно открыт файл, содержащий атрибутивную информацию, производится проверка описательной части атрибутивных полей. Если поля не соответствуют требованиям, пользователю предоставляется возможность осуществлять конвертацию без учета атрибутивной информации (в таком случае при обработе отдельных геометрических объектов будут использованы значения атрибутов по умолчанию). Логическое значение, возвращаемое функцией, указывает, прошел ли заголовок .MIF-файла проверку. По окончанию выполнения функции указатель .MIF-файла позиционируется на первой строке раздела геометрических данных об объектах.

`function IsMIFEof: Boolean` – функция, являющаяся локальной по отношению к функции `MIFHeaderProceed`. Возвращает истину в случае встретившегося признака конца файла. Отличается от стандартной функции `Eof` тем, что автоматически добавляет сообщение об ошибке в журнал выполнения обработки.

`procedure ConvertMIFFile` – Данная процедура производит выделение отдельных геометрических объектов .MIF-файла и связанных с ним атрибутов из .MID-файла. По типу объекта определяется, какая процедура будет вызвана для последующей конвертации. Поскольку формат MapInfo Interchange предусматривает возможность использования NONE-объектов (объектов, не имеющих геометрии, но обладающие атрибутикой), которые не будут иметь соответствующих им объектов в виртуальном мире, то такие файлы «обрабатываются» без вызова дополнительных процедур.

`procedure MakeVRMLRegion(Dtype: String; Dheight: Real; Dhdistance: Real; Dtexture: String; Durl: String; DParameters: String)` – процедура, генерирующая трехмерный объект виртуального мира на основе геометрических и атрибутивных данных о полигональных объектах. Осуществляет

чтение координат узлов полигона из .MIF-файла и использует атрибуты объекта, передаваемые в качестве параметров процедурой `ConvertMIFFile`. Моделирование трехмерного объекта осуществляется методом экструзии. В качестве многоугольника сечения используется сам полигон, его высота определяется либо параметром `Dheight`, либо используется значение по умолчанию. Поскольку в файлах обменного формата MapInfo Interchange координаты указываются относительно северо-западного угла, процедура выполняет их преобразование в координаты северо-восточного квадранта. Полученный объект смещается вдоль вертикальной оси в соответствии со значением параметра `Dheight`. Цвет формируемого объекта определяется из свойств кисти, указываемых геоинформационной системой в .MIF-файле. При необходимости добавляются дополнительные VRML-узлы, осуществляющие загрузку и наложение текстур из файла, указанного в параметре `Dtexture`, а так же осуществляющих переход при активации данного трехмерного объекта на указанный URL. Поле `DParameters` используется для произведения дополнительных трансформаций над текущим объектом. Если параметр `Dtype` имеет значение «ClearBorder», то процедура формирует полностью прозрачный объект, который не будет отображаться в VRML-сцене, но может быть использован в качестве дополнительных границ, не позволяющих пользователю заходить в определенные области.

`procedure MakeVRMLPoint(x, y: Real; Dtype: String; Dheight: Real; Dhdistance: Real; Dtexture: String; Durl: String; DParameters: String)` – данная процедура реализует встраивание в сцену виртуального мира VRML-объектов, представляющих отдельные точечные объекты. Подключение таких объектов производится методом Inlining. Имя встраиваемого файла должно быть указано в значении атрибута `Dtype`. Центр объекта помещается в точку (x, y). При этом объект может быть смещен вдоль вертикальной оси на значение параметра `Dhdistance`. Кроме визуальных объектов данная процедура может размещать на VRML-сцене так называемые точки обзора (Viewpoints). Для задания такой точки значение параметра `Dtype` должно содержать ключевое слово «Viewpoint». Координаты пользователя, помещенного в соответствующую точку обзора, задаются на плоскости координатами (x, y) а его высота – параметром . Дополнительные свойства точки обзора (угол обзора, дальность видения) определяются из значения параметра `DParameters`.

`function RealToStr(Val: Real): String` – дополнительная функция, преобразовывающая вещественные значения в текстовую строку с фиксированными параметрами. Используется процедурами `MakeVRMLRegion` и `MakeVRMLPoint` для помещения вещественных значений в VRML-код.