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

УДК 681.03

ДОПУСТИТЬ К ЗАЩИТЕ В ГАК
Зав. кафедрой, проф., д.т.н.
_____ Ю.Л. Костюк
“___”_____2003 г.

Кряжев Василий Владимирович

# Создание трехмерных моделей местности в среде ГИС ArcView

Дипломная работа

Научный руководитель,
ст. преп. кафедры теоретических
основ информатики *С.Ф. Трофимова*

Исполнитель,
студ. гр. 1481 *В.В. Кряжев*



Томск-2003

# Реферат

Дипломная работа на 77 страницах, 9 источников, 26 рисунков

ГЕОИНФОРМАЦИОННАЯ СИСТЕМА (ГИС), 2D ЦИФРОВЫЕ МОДЕЛИ, VRML, 3D-СЦЕНА, ЦИФРОВАЯ МОДЕЛЬ РЕЛЬЕФА (ЦМР), УСЛОВНЫЕ ЗНАКИ, ARCVIEW, DELPHI, ДИНАМИЧЕСКИЙ ОБМЕН ДАННЫМИ (DDE).

1. Объект исследования – цифровая модель местности.
2. Цель исследования – создание 3D моделей элементов картографического изображения.
3. Методы исследования – экспериментальный (на ЭМВ).
4. Полученные результаты – разработано приложение на Delphi для конструирования описания VRML-объектов, которые могут использоваться при построении трехмерной модели городских территорий.
5. Область применения – подразделения городской администрации (Городской департамент архитектуры и градостроительства и т.д.).

## Содержание

# Введение

Распространение компьютерных информационных технологий в наше время повлекло за собой развитие и географических информационных технологий. В течение более чем трех десятилетий разрабатывались двумерные ГИС, в которых географические данные представляются набором плоских объектов (точечные объекты - отдельно стоящие деревья, кусты и т.п.; линейные объекты - дороги, магистрали, электрические линии, трубопроводы; полигональные объекты - массивы лесов, поля и т.п.). Но по мере того, как эта технология все шире проникает в «массы», она неизбежно выходит за рамки плоскости. К недостаткам двумерных цифровых карт можно отнести сложную систему условных обозначений. Для простого пользователя это может стать серьезным препятствием на пути ознакомления с конкретной географической информацией. ГИС-технология позволяла получить достойные результаты во многом за счет того, что ее инструментарий использовался «экспертами». Создание 3D-моделей удобный способ передачи информации, поскольку объект может иметь на экране компьютера такой же вид, как и в реальном мире. Несмотря на то, что 3D визуализация очень актуальна, современные пользователи ГИС часто ограничены возможностями двумерного представления своих данных. Это происходит не из-за отсутствия трехмерных данных в системах - 3D атрибуты очень часто присутствуют в базах данных ГИС и доступны для широкого использования. Проблема состоит в существующих инструментальных средствах. Сегодняшний инструментарий обеспечивает пользователей сравнительно простыми способами создания и взаимодействия с трехмерными ГИС, однако большинство из них требуют высокого уровня технической подготовки и детального знания 3D концепций. В результате этого использование информации, содержащейся в ГИС для создания трехмерных сцен на сегодняшний день признается трудоемким и дорогостоящим процессом, а получаемое визуальное качество - невысоким и слабо интерактивным, недоступным широкому кругу пользователей.

Целью данной работы является расширение возможностей ГИС ArcView для создания 3D моделей, представляющих природный ландшафт, природные и техногенные объекты (например, деревья, здания, памятники, водные объекты и прочее) на основе цифровых 2D моделей с использованием языка VRML. Реализация такой возможности предполагает наличие данных - все исходные данные для построения 3D сцены находятся внутри ArcView GIS в виде слоев (за исключением подготовленной библиотеки).

VRML появился в связи с требованиями новых презентационных технологий, включающих мультимедиа и ее предельное состояние - виртуальную реальность. Использование для визуализации VRML позволяет не только получать 3D сцены, но и широко их представлять в сети Интернет/Интранет. Просмотр модели возможен стандартными средствами – обычным интернет-браузером с установленным плагином VRML. Использование этого широко распространенного формата не накладывает ограничений на компьютер.

# 1. Карты как пространственные модели

Карта – это модель местности, описывающая пространственно-временные отношения объектов и явлений относительно земной поверхности.

Карты допускают единовременный обзор пространства в любых пределах – от небольшого участка местности до поверхности Земли в целом. Они создают зрительный обзор формы, величины и взаимного положения объектов, позволяют находить их пространственные размеры: координаты, длины, площади, высоты и объемы. Карты содержат необходимые количественные и качественные характеристики этих объектов и, наконец, показывают существующие между ними связи: пространственные и некоторые другие. Эти свойства объясняют значение и ценность карт для практики.

Термин «карта» происходит от греческого слова chartes, которое означало лист из папируса для письма. Издавна было принято определять географическую карту как уменьшенное изображение земной поверхности на плоскости. Но это определение неточно и неполно. Во-первых, оно справедливо и в отношении любого фотоснимка земной поверхности, и в отношении пейзажа - изображения местности средствами изобразительного искусства. Во-вторых, оно ограничивает задачи карты изображением земной поверхности, тогда как современные карты включают в свое содержание самые разнообразные природные и социально-экономические явления (например, температуру и давление воздуха, национальный состав населения и др.), правда, всегда показываемые относительно земной поверхности.

Три черты определяют специфику географических карт: 1) математически определенное построение; 2) использование особых знаковых систем (картографических символов); 3) отбор и обобщение изображаемых [1].

Математически определенное построение карт предусматривает установление строгой функциональной зависимости между географическими координатами точек земной поверхности и прямоугольными координатами тех же точек на плоскости. Это позволяет изучать по картам пространственные отношения и формы изображаемых объектов.

Развернуть поверхность эллипсоида на плоскости без складок и разрывов (или без сжатия и растяжения) невозможно. Земную поверхность нельзя изобразить на плоскости без геометрических деформаций, называемых искажениями, сохраняя истинные горизонтальные очертания объектов. Поэтому для перехода от поверхности эллипсоида к плоскости используют математические способы отображения, называемые картографическими проекциями, которые устанавливают определенную функциональную зависимость между координатами точек на эллипсоиде и плоскости. Когда такая зависимость известна, можно учитывать деформации плоского изображения и определять по карте с необходимой точностью расстояние, площади и углы.

Географические карты передают явления с помощью картографических знаков - специальных графических символов. Эти знаки создают на карте пространственный образ действительности. Смысл применения картографических знаков становится понятным при сопоставлении карты с аэрофотоснимком той же местности. Картографические знаки «стирают» многие индивидуальные черты объектов местности и тем самым обедняют изображение.

Использование картографических знаков позволяет:

а) сильно уменьшать изображение земной поверхности, чтобы охватить единым взглядом необходимую часть или даже всю земную поверхность, воспроизводя при этом на карте те объекты местности, которые в силу уменьшения не выражаются в масштабе карты, но по своему значению должны быть показаны;

б) показывать на карте рельеф земной поверхности, т. е. передавать неровности в плоском изображении;

в) не ограничиваться отображением на карте только внешнего вида предметов (явлений), а указывать их внутренние свойства;

г) показывать распространение явлений, не воспринимаемых нашими органами чувств (например, температуру и т. п.), и делать наглядными недоступные непосредственному восприятию связи и отношения (например, связи между источниками сырья и предприятиями по его переработке);

д) исключать малозначимые стороны, частности и детали, свойственные единичным предметам (явлениям), и выделять их общие и существенные признаки, т. е. прибегать к абстракции.

**Определение.** *Географическими картами* называют математически определенные, обобщенные образно-знаковые изображения земной поверхности на плоскости, показывающие размещение, состояние и связи различных природных и общественных явлений, отбираемых и характеризуемых в соответствии с назначением каждой конкретной карты.

Географические карты принадлежат к пространственным образно-знаковым моделям - они используют язык знаков и дают пространственный образ отображаемых явлений. Вместе с тем карты можно относить к мысленным моделям, поскольку для пользования ими необходимо понимание знаков и мысленное пространственное восприятие изображенной на карте действительности.

Для создания и использования географических карт необходимо знать их свойства и особенности. В карте различают картографическое изображение, математическую основу, вспомогательное оснащение и дополнительные данные.

Картографическое изображение - главная часть любой географической карты - заключает в себе некоторую совокупность сведений (информацию) о показанных на карте природных и социально-экономических объектах (явлениях), их размещении, свойствах, связях. Эти сведения составляют содержание карты, которое может быть разделено на отдельные географические элементы по однородным группам показываемых на карте объектов. Комплекс элементов содержания неодинаков на разных картах. Так, главными элементами содержания тематических карт могут быть полезные ископаемые, почвы, животный мир и др. Но один элемент, а именно воды (берега морей и озер, речная сеть), желателен для всякой карты; он важен для привязки других элементов ее содержания.

При анализе картографического изображения следует различать заключенное в нем содержание и форму передачи этого содержания посредством определенной системы картографических знаков и надписей.

Геометрические законы построения и геометрические свойства картографического изображения определяются его математической основой, к элементам которой принадлежат картографическая проекция и связанная с ней координатная сетка, масштаб и опорная геодезическая сеть.

Картографическая проекция, выражающая аналитическую зависимость между координатами точек поверхности земного эллипсоида и его изображения на плоскости, обязывает начинать работы по созданию карты с построения системы координатных линий - плоского изображения соответствующих линий на поверхности эллипсоида. Та или иная координатная сетка лежит в основе всякой географической карты, т. е. относится к обязательным ее элементам. Однако на некоторых картах координатные сетки отсутствуют, что допустимо, когда карта покрывает относительно небольшое пространство, не предназначена для измерений или является схемой.

Кроме картографического изображения всякая карта имеет вспомогательные элементы, облегчающие чтение карты и работу с ней. К вспомогательным элементам принадлежат легенда карты - таблица картографических знаков (условных обозначений) с необходимыми пояснениями и графики для измерений по картам (расстояний, углов, площадей, координат точек и т. п.). В легенде важны: полнота, т. е. включение всех использованных на карте знаков; ясность и, по возможности, краткость поясняющих знаки текстов; логичность в группировке и размещении знаков. Правильно построенная легенда раскрывает содержание карты - перечень элементов, классификации и показатели, использованные для каждого элемента, а также степень их обобщения. Легенда помещается на полях карты или на свободных пространствах внутри ее рамки. Для многолистных карт легенду иногда печатают на отдельном листе или в виде брошюры.

К вспомогательным элементам относят и название карты, фамилии автора и редактора, справочные данные о времени составления карты, об использованных источниках и др., а на изданных картах также выходные данные - название издательства, место и год издания и т. п.

На полях карты или ее свободных местах внутри рамки иногда помещают дополнительные карты и графические построения (профили, диаграммы и т. п.), таблицы и текстовые данные, которые поясняют, дополняют и обогащают в том или ином отношении картографическое изображение.

Применяемые на картах обозначения различных объектов и их качественных и количественных характеристик называют картографическими условными знаками. Картографические знаки – язык карты. Они передают ее содержание, т. е. совокупность заключенных в карте знаний о реальной действительности. Картографические знаки – специальные графические символы – обозначают на карте предметы, явления, процессы. Их используют для реальных и абстрактных объектов. Они должны быть по возможности простыми, экономичными по занимаемой ими площади и при этом должны четко отличаться друг от друга и легко опознаваться.

С точки зрения передачи на карте плановых геометрических особенностей объектов различают внемасштабные, площадные и линейные условные знаки.

*Внемасштабные условные знаки* применяются для изображения объектов, площади которых не выражаются в масштабе карты или выражаются на ней столь малым размером, что не могут ясно различаться, При этом очертания объектов, как правило, в обозначении не сохраняются.

*Площадные условные знаки* применяются для заполнения площадей объектов, выражающихся в масштабе карты, при этом очертания объектов на карте сохраняются. Они строятся с помощью фоновой окраски или других площадных графических средств.

*Линейные условные знаки* применяются для изображения объектов линейного характера, длина которых выражается в масштабе карты. При этом сохраняется подобие линейных очертаний, но часто преувеличивается их ширина.

При построении картографических обозначений используется различные графические средства. Простейшими из них являются точки, линии, штрихи. Они лежат в основе более сложных графических средств, которые условно можно разделить на четыре типа:

- знаковые – фигурные и геометрические знаки;
- линейные – одинарные, двойные, тройные линии, полосы, стрелы и т.п.;
- площадные – штриховки, фоновые окраски, равномерно покрывающие площадь значка какого-либо рисунка, разные оттенения или окраски оконтуривающей площадь линии с ее внутренней стороны, различные индексы (буквенные и др.), помещаемые внутри оконтуренной площади, растянутые в пределах площади надписи;
- буквенные и цифровые – отдельные буквы, сокращенные или полные слова, различные числа и др.

Графическими средствами, позволяющими разнообразить картографические обозначения, являются также их величина, форма, ориентировка, внутренний рисунок, цвет, его насыщенность и светлота. Такое средство, как полутоновое изображение, дающее постепенный переход цвета от светлого к темному, используется на плоской карте для передачи объемов каких-либо объектов, например форм рельефа.

Кроме плановых геометрических свойств различные объекты, явления и образуемая ими в совокупности реальная географическая действительность в целом имеют и ряд других особенностей, в том числе количественных и качественных, динамических и пространственных. Для визуализации на картах этих особенностей используют разные способы изображения.

*Способ значков* применяется для изображения объектов и явлений, локализованных по пунктам с помощью тех или иных значков, размеры которых принимаются постоянными или меняются по какой либо шкале и которые помещаются на карте по месту нахождения самих объектов. Различают значки трех видов. Геометрические фигуры которые имеют форму прямоугольника, круга или другой простой фигуры. Буквенные значки – это одна или несколько начальных букв названия изображаемого объекта или явления. Наглядные значки своим видом напоминают изображаемые объекты или явления. Они бывают натуралистические или символические. Форма, внутренний рисунок или цвет значка обычно отражает качественные особенности объекта или явления, а его размер – количественную характеристику.

*Способ локализованных диаграмм* – способ изображения на карте явлений, имеющих сплошное или линейное распространение, с помощью графиков или диаграмм, показывающих явление в местах его изучения. Например, изображение изменения температуры воздуха и количества осадков по месяцам года с помощью кривых, показывающих это изменение в местах нахождения метеостанций.

*Способ изолиний* - способ изображения явлений, обычно имеющих сплошное распространение, с помощью кривых линий, соединяющих на карте точки с одинаковым значением какого-либо количественного показателя явления. Нередко для наглядного показа изменения величины явления в пределах изображаемой территории полосы между изолиниями окрашивают по цветовой шкале.

*Способ качественного фона* – способ, изображения на карте качественных различий какого-либо явления в пределах изображаемой территории путем деления ее на части и покрытия каждого из них с помощью одного из площадных графических средств.

При этом деление территории на части органически связано с отображаемым явлением. Способ качественного фона используется как основной на картах различного вида районирования, картах народов, геологических, ботанических и др.

*Способ количественного фона* – способ изображения на карте количественных различий какого-либо явления в пределах изображаемой территории путем деления ее на части и покрытия каждой из них с помощью одного из площадных графических средств. При этом деление территории на части органически связано с отображаемым явлением. Границы между этими частями проводят по признакам, связанным с отображаемым явлением, и для каждой части по тем или иным данным указывают количественную характеристику отображаемого явления.

*Способ ареалов* – способ изображения на карте области ограниченного по площади распространения какого-либо явления с помощью того или иного площадного графического средства. При обозначении ареала растянутой надписью или заполнением его площади равномерно покрывающими значками сама граница ареала нередко не показывается. Различают абсолютные ареалы, за пределами которых явление отсутствует и относительные ареалы, выделяемые или по преобладанию явления, или по его особым свойствам.

*Точечный способ* – способ изображения на карте рассредоточенных объектов множеством точек одинакового размера, обозначающих одинаковое количество единиц изображаемого объекта и располагаемых соответственно его размещению и концентрации. Перед картографированием устанавливают «вес» точки и определяют, какой величине количественного показателя соответствует одна точка. Точки располагают на карте так, чтобы они наилучшим образом отражали действительное размещение объекта или явления. Иногда используют точки двух размеров. Точечный способ дает наглядную картину распространения объекта или явления в пределах изображаемой территории. «Вес» точки позволяет определить его количественную характеристику.

*Способ линейных знаков* – способ изображения на карте различных линейных объектов, практически не имеющих ширины, объектов линейного протяжения, ширина которых не выражается в масштабе карты и линий протяженности вытянутых объектов. Для передачи качественных и количественных характеристик используют рисунок, цвет, структуру линейных знаков, а иногда и ширину.

*Способ знаков движения* – способ изображения на карте различных пространственных перемещений. Этим способом могут быть показаны линии или полосы, по которым происходит перемещение, направление, количество, скорость перемещения и другие данные. В качестве графических средств используются стрелки разного цвета, рисунка и ширины.

*Картодиаграмма* – способ изображения суммарной величины какого-либо явления в каждой единице территориального деления с помощью диаграммных фигур, выражающих эту величину и помещаемых внутри каждой такой единицы. Причем территориальное деление не связано прямо с отображаемым явлением. Карта в целом показывает распределение величины данного явления в пределах изображаемой территории. Способ картодиаграммы чаще всего используется на картах, составленных по статистическим данным, относящимся к единицам территориального деления.

*Картограмма* – способ изображения на карте средней интенсивности какого-либо явления в каждой единице территориального деления с помощью одного из площадных графических средств, например, фоновой окраски или штриховки, интенсивность которых обычно соответствует интенсивности данного явления. Причем территориальное деление

не связано прямо с отображаемым явлением. Карта в целом показывает изменение интенсивности данного явления в пределах изображаемой территории, Способ картограммы чаще всего используется на картах, составленных по статистическим данным, относящимся к единицам территориального деления.

Часто для отображения какого-либо явления используют сочетание разных способов.

В последние несколько лет в компьютерной картографии начало формироваться новое направление - виртуальное моделирование и картографирование. Английское слово “virtual” означает фактический, действительный, в смысле, близком к слову “реальный”. Визуализация виртуальной реальности опирается, прежде всего, на применение эффектов трехмерности и анимации. Именно они создают иллюзию присутствия и перемещения в объемном пространстве. При этом реализуются четыре главных свойства:

- сочетание в одном геоизображении свойств карты, перспективного снимка, блок-диаграммы и анимации;
- возможность программного управления этим синтезированным геоизображением;
- интерактивное взаимодействие с самим геоизображением и окружающей его виртуальной средой;
- уменьшение свойств знаковости и условности геоизображения, придание ему реалистических черт.

# 2. Географическая информационная система

## 2.1. Что такое ГИС

Географическая информационная система (ГИС) - это возможность нового взгляда на окружающий нас мир. ГИС - это современная компьютерная технология для отображения и анализа объектов реального мира, также событий, происходящих на нашей планете. Эта технология объединяет традиционные операции работы с базами данных, такими как запрос и статистический анализ, с преимуществами полноценной визуализации и географического (пространственного) анализа, которые предоставляет карта. Эти возможности отличают ГИС от других информационных систем и обеспечивают уникальные возможности для ее применения в широком спектре задач, связанных с анализом и прогнозом явлений и событий окружающего мира, с осмыслением и выделением главных факторов и причин, а также их возможных последствий. Создание карт и географический анализ не являются чем-то новым. Однако технология ГИС предоставляет новый, более современный, более эффективный, удобный и быстрый подход к анализу проблем и решению задач, стоящих перед человечеством в целом, и конкретной организацией или группой людей, в частности. Она автоматизирует процедуру анализа и прогноза. До начала применения ГИС лишь немногие обладали искусством обобщения и полноценного анализа географической информации с целью обоснованного принятия оптимальных решений, основанных на современных подходах и средствах.

В настоящее время ГИС - это многомиллионная индустрия, в которую вовлечены сотни тысяч людей во всем мире. Эту технологию применяют практически во всех сферах человеческой деятельности - будь то анализ таких глобальных проблем как перенаселение, загрязнение территории, сокращение лесных угодий, природные катастрофы, так и решение частных задач, таких как поиск наилучшего маршрута между пунктами, подбор оптимального расположения нового офиса, поиск дома по его адресу, прокладка трубопровода на местности, различные муниципальные задачи.

## 2.2. Составные части ГИС

ГИС включает в себя пять ключевых составляющих: аппаратные средства, программное обеспечение, данные, исполнители и методы.

Аппаратные средства. Это компьютер, на котором запущена ГИС. В настоящее время ГИС работают на различных типах компьютерных платформ, от централизованных серверов до отдельных или связанных сетью настольных компьютеров.

Программное обеспечение ГИС содержит функции и инструменты, необходимые для хранения, анализа и визуализации географической (пространственной) информации. Ключевыми компонентами программных продуктов являются: инструменты для ввода и оперирования географической информацией; система управления базой данных (СУБД); инструменты поддержки пространственных запросов, анализа и визуализации (отображения); графический пользовательский интерфейс (GUI) для легкого доступа к инструментам.

Данные - это вероятно наиболее важный компонент ГИС. Данные о пространственном положении (географические данные) и связанные с ними табличные

данные могут собираться и подготавливаться самим пользователем, либо приобретаться у поставщиков на коммерческой или другой основе. В процессе управления пространственными данными ГИС интегрирует пространственные данные с другими типами и источниками данных, а также может использовать СУБД, применяемые многими организациями для упорядочивания и поддержки имеющихся в их распоряжении данных.

Исполнители. Широкое применение технологии ГИС невозможно без людей, которые работают с программными продуктами и разрабатывают планы их использования при решении реальных задач. Пользователями ГИС могут быть как технические специалисты, разрабатывающие и поддерживающие систему, так и обычные сотрудники (конечные пользователи), которым ГИС помогает решать текущие каждодневные дела и проблемы.

Методы. Эффективность (в том числе экономическая) применения ГИС во многом зависит от правильно составленного плана и правил работы, которые составляются в соответствии со спецификой задач и работы каждой организации.

## 2.3. Как работает ГИС

ГИС хранит информацию о реальном мире в виде набора тематических слоев, которые объединены на основе географического положения. Любая географическая информация содержит сведения о пространственном положении, будь то привязка к географическим или другим координатам, или ссылки на адрес, почтовый индекс, идентификатор земельного или лесного участка, название дороги и т.п. При использовании подобных ссылок для автоматического определения местоположения объекта (объектов) применяется процедура, называемая геокодированием. С ее помощью можно быстро определить и посмотреть на карте где находится интересующий вас объект или явление, такие как дом, в котором проживает ваш знакомый или находится нужная вам организация, где произошло землетрясение или наводнение, по какому маршруту проще и быстрее добраться до нужного вам пункта или дома.

## 2.4. 2D и 3D ГИС, их сходства и различия

Распространение компьютерных информационных технологий в наше время повлекло за собой развитие и географических информационных технологий. Еще недавно в геоинформационных системах, как правило, применялись двумерные пространственные данные, в которых географические данные представляются набором плоских объектов трех типов (точечные объекты - дома, отдельно стоящие деревья, кусты и т.п.; линейные объекты - дороги, магистрали, электрические линии, трубопроводы; полигональные объекты - массивы лесов, поля и т.п.) и условных обозначений, идентифицирующих различные реальные объекты. Сейчас ГИС в основном работают в так называемом 2,5-мерном пространстве, когда величина Z атрибутивно привязана к точке (X,Y), часто через цифровые модели рельефа. С недавнего времени осуществляется переход к полноценным трехмерным ГИС.

Потребность в реалистичном отображении окружающего мира увеличивает значимость трехмерного (3D) моделирования. 3D модели облегчают планирование, контроль и принятие решений во многих отраслях. Трехмерная визуализация территорий методами компьютерной графики и создание муниципальных трехмерных ГИС способны

изменить технологию и практику управления городом, городского планирования окружающей среды, разработки и ведения проектов.

Однако современные аппаратные решения не позволяют реализовать 3D ГИС в полном объеме и визуализируют географические данные более схематично. Лишь специализированные графические рабочие станции Silicon Graphics, Sun Ultra и другие могут справится с такого рода задачами как визуализация данных. Именно на этих рабочих станциях реализованы наиболее используемые 3D ГИС.

Основное отличие между 2D и 3D ГИС, конечно, это добавление дополнительной пространственной координаты, которое также определяет отличие в некоторых функциях. Сходств этих двух технологий очень много, поскольку основной частью геоинформационных систем являются географические данные, а не их представление. Поэтому основное сходство - это одинаковая модель данных. Только в трехмерном случае добавляются атрибуты и связи ответственные за третью координату. Также как и в 2D в 3D существует послойное разбиение данных на типы. В трехмерном случае пространственная визуализация географической информации позволяет просматривать гораздо больше. Функцию скрытия с вида слоев в 3D системе играет роль прозрачности определенных типов объектов. Например, просмотр подземных коммуникаций или линий метро в 3D случае может представлять проблему, если не реализовать данную функцию.

Двумерные ГИС бывают и векторные и растровые, трехмерные ГИС бывают векторные и воксельные.

Трехмерная ГИС гораздо сложнее в разработке и требует очень больших аппаратных ресурсов. Стандартный персональный компьютер не может справиться с такими задачами, что и ограничивает их применение. Рассмотрим основные функции ГИС и методы их реализации.

Построение изображения в 2D ГИС реализуется довольно просто. В рамках одного слоя строятся линии и полигоны, используя различные типы линий или типы закрасок. Ограничения целостности в таких системах - проверка на висячие вершины, на пересечение объектов, в том числе и из разных слоев, проверка некоторых типов объектов, например зданий, на перпендикулярность. Вычисление линейных размеров или расстояния между двумя точками не представляют затруднений.

Теперь рассмотрим реализацию 3D ГИС. Построение трехмерного изображения в любой программе реализуется непросто. В 3D ГИС должно быть реализовано множество функций работы с трехмерными объектами. Основная функция - показ изображения с разных точек и плавные переходы между ними. Ограничения целостности в данных системах такие же, как и в 2D системах, только в трехмерном варианте. Вычисление расстояния между двумя точками довольно проблематично, так как задавать конечные точки с нужной точностью с помощью указателя мыши сложно. Вообще производить замеры и вычисления в трехмерном пространстве, который визуализируется на двумерной плоскости монитора представляет собой отдельную проблему.

## 2.5. Классификация 3D ГИС

Хотя трехмерные ГИС могут использоваться для карт любого масштаба и назначения, на данный момент программное обеспечение 3D ГИС используется только для узких геоинформационных целей. Рассмотрим самые важные из них:

1. ***Формирование рельефа поверхности.***

Это наиболее простая и часто используемая технология. Она применяется не только в специализированных геосистемах, но и в многочисленных демо программах, играх и симуляторах. В основном, в качестве данных для этого класса 3D ГИС используется матрица высот и вид поверхности. Также дополнительно могут задаваться мелкие объекты типа деревьев, домов и т.п. Показ особенностей рельефа является основной функцией. Это реализуется путем пролета камеры над поверхностью земли на высоте птичьего полета.

С помощью программ такого класса могут быть представлены как крупномасштабные, так и мелкомасштабные карты. Отличие - в степени генерализации реальных географических данных.

2. ***Трехмерные кадастровые системы и визуализация городских массивов.***

Эта технология получила наибольшее распространение в связи с большой практической необходимостью. Кадастровые системы и системы городского планирования и управления широко применяются в народном хозяйстве для разнообразных целей.

Для данного типа систем в основном используются крупномасштабные географические и топографические карты. Данными могут являться любые типы объектов в зависимости от детализации системы и ее конкретных целей : здания, дороги, различные коммуникации, лесные насаждения, любые мелкие объекты, такие как осветительные столбы и другие объекты используемые в городском хозяйстве. Соответственно основной функцией таких систем вообще, является получение разнообразной информации по каждому из объектов, возможность различных выборок и агрегации данных. А в трехмерных городских ГИС добавляется возможность удобного осмотра планировки и расположения самих объектов и другие функции присущие только трехмерным системам.

Для городских 3D ГИС одной из важнейших функций является плавное увеличение/уменьшение детализации изображения. Если разработчик 3D системы планирует реально отображать вид объектов, то необходимо хранить текстуры всех типов объектов и производить неискажающие преобразования с ними в реальном времени.

## 2.6. Обзор существующих решений 3D ГИС

Система визуализации CAD и ГИС данных **CLRView** разработанную *Centre for Landscape Research (CLR)***.** Этот программный пакет работает на рабочих станциях *Silicon Graphics IRIS.* Эта система поддерживает работу со многими известными форматами данных *DXF, TIN, DEM, Lattices, and Arc/Info Coverages,* и обеспечивает их прозрачную интеграцию. **CLRView** также позволяет работать со специальными атрибутами *Arc/Info* и автоматически генерировать трехмерные изображения из двумерных карт для массивов зданий, лесов и других объектов.

Существует интересная геоинформационная система для планирования городских массивов **NPAC**. После разработки архитекторами плана некоторого района эти данные вводятся в систему и формируется искусственное трехмерное изображение этого района. Затем по этому району можно прогуляться и оценить все достоинства и недостатки расположения зданий и других объектов.

Еще один пример реальной системы - **3-Dimensional Marine Geographical Information System of Prydz Bay.** Это трехмерная ГИС предназначена для управления морской и береговой антарктической экосистемой *Prydz Bay, Австралия*. Система позволяет отслеживать геологические и океанологические процессы, а также ареалы различных видов флоры и фауны.

Океанографические данные включают в себя температуру воды, океанические течения, ледовые массивы. Биологические данные включают пути миграции рыб и морских животных, их количество.

Система использует технологию визуализации научных данных Application Visualisation System (AVS). Данные хранятся в формате Arc/Info, а затем преобразуются в формат AVS. Для различных классов данных используются различные технологии и программы обработки. Вся эта система генерирует трехмерное изображение в реальном времени с использованием компьютеров *Silicon Graphics Indigo2 XZ.*

# 3. VRML - язык моделирования виртуальной реальности

VRML: аббревиатура от Virtual Reality Modeling Language. VRML одно из наиболее интересных направлений в WEB-технологии. VRML разрабатывался для использования в Internet, intranet, локальных вычислительных сетях. VRML файловый формат описывающий 3D объекты и миры. С помощью VRML можно создавать статические и динамические 3D и мультимедийные объекты с гиперссылками на другие объекты, такие как текст, звук, картинки, кино. Пользователи могут самостоятельно исследовать VRML миры.

VRML был разработан в соответствии со следующими требованиями:

- Обеспечить возможность использования, комбинирования динамических 3D объектов, их многократного использования в пределах VRML мира;
- Обеспечить возможность добавлять новые объектные типы не определенные в VRML;
- Обеспечение аппаратной независимости;
- Реализация на широком ряду компьютерных платформ;
- Поддержка динамических 3D миров произвольных размеров;
- Позволить разработку компьютерных программ способных создавать, редактировать, поддерживать VRML файлы, как, например программы конвертирующие из других 3D файловых форматов в VRML файлы.

## 3.1. Система координат VRML

Система координат VRML основана на типичном математическом представлении трехмерных пространств – это правая система координат, в которой точки проецируются на x-, y- и z- оси (система Декартовых координат).

Значения по оси x увеличиваются слева направо, по оси y - снизу вверх и по оси z – по направлению к наблюдателю: чем больше значение по оси z, тем ближе точка расположена к наблюдателю. Длина и расстояние в этой системе измеряются в метрах, а углы в радианах. По умолчанию изображение объекта представляет собой проекцию объекта на плоскость xy.

## 3.2. Ориентация

При использовании VRML-браузера можно выбрать способ ориентации сцены и точку наблюдения . VRML- браузеры предлагают различные режимы перемещения: “прогулки”, ”исследования”, ”полета”. Кроме того, можно просматривать сцены из разных точек наблюдения.

## 3.3. Точки наблюдения

При создании VRML- мира можно выбрать точку наблюдения (Viewpoint), т.е. положение (идеально подобранное для данной сцены), из которого пользователи могут просматривать мир.

## 3.4. Источники освещения

Глаза человека не видят в полной темноте, это касается и VRML миров. Единственный способ увидеть объект в VRML-мире - это осветить его. Добавляя источники освещения, можно создать эффекты затенения, блеска, бликов подобно тому, как это происходит в реальном мире.

В виртуальных мирах имеется источник освещения, часто упоминаемый как *headlight* (базовый источник освещения), который VRML- браузер автоматически использует в каждом мире. Headlight перемещается вместе с вами, будто он является вашем фонарем на шлеме. Но свет источника headlight распространяется только вперед и на небольшое расстояние; его недостаточно для эффективного освещения VRML-миров. Источники освещения не обладают формой и никогда не будут видны, они просто создают эффект освещения миров, воспринимаемый из точки наблюдения.

## 3.5. Рендеринг (визуализация)

В процессе визуализации воспроизводится сцена по содержимому графической база данных. Это обозначит, что выбирая в браузере тип рендеринга вы сообщаете компьютеру, каким образом необходимо отобразить графические объекты.

Большинство VRML-браузеров имеет меню рендеринга, с помощью которого можно выбрать желаемый способ отображения сцены. Необходимо учитывать зависимость качества визуализации от количества потребляемых вычислительных ресурсов и скорости визуализации.

## 3.6. Структура VRML-файла

VRML использует набор объектов, называемых узлами, которые применяются для обработки мультимедийных данных, создания интерактивной среды, а также для реализации трехмерной графики. Все узлы хранят данные в специальных областях - полях (fields) и событиях (events).

Узлы описывают формы объектов и свойства этих форм в VRML-мирах. Когда браузер читает VRML- файл, он следуя инструкциям, поступающим от узлов, создает VRML-мир.

Существует три основных класса узлов:

- Property (свойства);
- Shape (форма);
- Group (группа);

Для хранения необходимых данных в узлах выделяются специальные поля. Узлы и их поля определяют форму и цвет отображаемых объектов, местоположение изображения, а также другие свойства, используемые при построении виртуального мира, описанного в VRML-файле. Все элементы графа описания сцены являются узлами.

## 3.7. Структура узла

Узлы содержат следующую информацию:

- Имя типа узла.
- Параметры, называемые полями (fields). Информация, содержащаяся в полях, позволяет отличить конкретный узел от других узлов того же типа.
- Набор событий, которые данный узел может передавать или принимать.

Обязательными частями этой конструкции являются только имя типа узла и фигурные скобки.

## 3.8. Поля

Поля содержат внутренние характеристики узла, которые придают ему уникальные свойства. Узлы могут не иметь полей вообще или иметь их очень много.

В описании узла для каждого из полей определены: тип, имя, значение по умолчанию. Каждая характеристика узла (размер, цвет, местоположение и т.д.) определяется количественным значением. Если значение поля не определено, то используется значение по умолчанию. Порядок, в котором располагаются описания полей, не имеет значения.

Поля делятся на два класса: Single-Value и Multiple-Value. Поля Single-Value (имена типов начинаются с символов “SF”) хранят только одно значение: число, вектор. Поля Multiple-Value (имена типов начинаются с символов “MF”) могут содержать множество элементов данных. Обычно вся группа данных заключается в квадратные скобки, а отдельные величины отделяются друг от друга пробелами.

## 3.9. Группирующие узлы и узлы листья

Существует несколько различных классов узлов, но большинство попадает в две категории: группирующие узлы и узды листья. Группирующие узлы позволяют создавать списки узлов (коллекции). С коллекцией можно работать в дальнейшем как с самостоятельным объектом. Группирующие узлы дают возможность включать одни группы в состав других групп.

У узлов листьев не может быть потомков. Узлы этого типа могут включаться в группы.

## 3.10. Тиражирование объектов

Если геометрическая форма используется несколько раз в рамках одного и того же VRML-файла, то говорят, что происходит тиражирование объектов. Тиражирование позволяет использовать именованные узлы независимо от их сложности или размера. Все получающиеся в результате тиражирования объекты идентичны и различаются лишь ориентацией в пространстве. Тиражирование «сложных» объектов может заметно уменьшить размер VRML-файла, что позволяет экономить время на передачу этого файла по низкоскоростным каналам связи, время необходимое для визуализации сцены тоже заметно уменьшается.

## 3.11. Обзор типов узлов (VRML 2.0)

Здесь будут перечислены не все стандартные узлы, а только те которые автор использовал в своем проекте. Структура описания узлов будет следующей:

<тип_поля> <тип_значение_поля> <имя_поля> <значение_по_умолчанию>

**Transform {**

| | | | |
|---|---|---|---|
| eventIn | MFNode | addChildren | |
| eventIn | MFNode | removeChildren | |
| exposedField | SFVec3f | center | 0 0 0 |
| exposedField | MFNode | children | [ ] |
| exposedField | SFRotation | rotation | 0 0 1 0 |
| exposedField | SFVec3f | scale | 1 1 1 |
| exposedField | SFRotation | scaleOrientation | 0 0 1 0 |
| exposedField | SFVec3f | translation | 0 0 0 |
| field | SFVec3f | bboxCenter | 0 0 0 |
| field | SFVec3f | bboxSize | -1 –1 -1 |

**}**

Узел Transform (преобразования) является группирующим узлом, определяющим систему координат для своих потомков относительно родительской системы координат. Группирующие узлы могут выполнять только одно преобразование системы координат. Поля *translation, rotation, scale* служат соответственно для *переноса, поворота, масштабирования* системы координат.

**Group {**

| | | | |
|---|---|---|---|
| eventIn | MFNode | addChildren | |
| eventIn | MFNode | removeChildren | |
| exposedField | MFNode | children | [ ] |
| field | SFVec3f | bboxCenter | 0 0 0 |
| field | SFVec3f | bboxSize | -1 –1 –1 |

**}**

Узлы Group (группа) аналогичны узлам Transform, за исключением того, что в них отсутствуют поля, описывающие преобразования.

**Collision {**

| | | |
|---|---|---|
| eventIn | MFNode | addChildren |
| eventIn | MFNode | removeChildren |

| | | | |
|---|---|---|---|
| exposedField | MFNode | children | [ ] |
| exposedField | MFNode | collide | TRUE |
| field | SFVec3f | bboxCenter | 0 0 0 |
| field | SFVec3f | bboxSize | -1 –1 –1 |
| field | SFNode | proxy | NULL |
| eventOut | SFTime | collideTime | |

}

Используя узлы типа Collision, браузер может обнаружить конфликты между геометрическими представлениями «виртуального двойника» и другими объектами виртуального мира. Обнаружив конфликт, браузер предотвращает столкновение.

**Shape {**

| | | | |
|---|---|---|---|
| field | SFNode | appearance | NULL |
| field | SFNode | geometry | NULL |

}

Узлы Shape (форма) содержат информацию о геометрической форме и внешнем виде объектов. Узел Shape содержит только два поля. Поле geometry хранит тип геометрической формы (Sphere, Cone, IndexedFaceSet и т.д.), а поле appearance содержит ссылку на узел типа Appearance.

**Appearance {**

| | | | |
|---|---|---|---|
| exposedField | SFNode | material | NULL |
| exposedField | SFNode | texture | NULL |
| exposedField | SFNode | textureTransform | NULL |

}

С помощью узла Appearance задаются визуальные свойства геометрических объектов. Для этого имеются два поля, в которых хранятся ссылки на узлы типа Texture и Material. Каждое из этих полей может иметь значение NULL. Если значение в поле material определено, то оно содержит имя узла типа Material. Значение поля texture задает тип узла текстуры (PixelTexture, ImageTexture, MovieTexture) .

**ImageTexture {**

| | | | |
|---|---|---|---|
| exposedField | MFString | url | [ ] |
| field | SFBool | repeatS | TRUE |
| field | SFBool | repeatT | TRUE |

}

Узел ImageTexture содержит поле url, определяющего адреса (Uniform Resource Location) файла содержащего текстуру. Поля repeatS, repeatT определяют тиражирование текстуры.

**PixelTexture {**

| | | | |
|---|---|---|---|
| exposedField | SFImage | image | 0 0 0 |
| field | SFBool | repeatS | TRUE |
| field | SFBool | repeatT | TRUE |

}

Узлы PixelTexture используются для задания текстур, основанных на двухмерных изображениях. Поле image используется для явного задания цвета каждого пиксела.

**Materail {**

| | | | |
|---|---|---|---|
| exposedField | SFFloat | ambientIntensity | 0.2 |
| exposedField | SFColor | diffuseColor | 0.8 0.8 0.8 |
| exposedField | SFColor | emissiveColor | 0 0 0 |

| | | | |
|---|---|---|---|
| exposedField | SFFloat | shininess | 0.2 |
| exposedField | SFColor | specularColor | 0 0 0 |
| exposedField | SFFloat | transparency | 0 |

}

Узлы типа Material определяют свойства материала поверхности для узлов типа Shape. Узлы Material позволяют определить существенные свойства поверхности – цвет, способность объекта поглощать или отражать свет, также можно определить прозрачность объекта.

**Box {**

| | | | |
|---|---|---|---|
| field | SFVec3f | size | 2 2 2 |

}

Узел Box используется для задания объекта в форме прямоугольного параллелепипеда. Поле size определяет размеры параллелепипеда (см. рис 3.1).

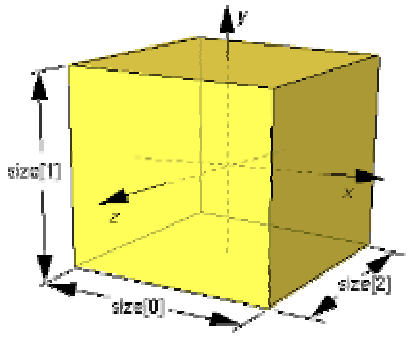



Рисунок 3.1 Параметры, задающие параллелепипед

**Cone{**

| | | | |
|---|---|---|---|
| field | SFFloat | bottomRadius | 1 |
| field | SFFloat | height | 1 |
| field | SFBool | side | TRUE |
| field | SFBool | bottom | TRUE |

}

Узел Cone описывает конус (см. рис. 3.2).

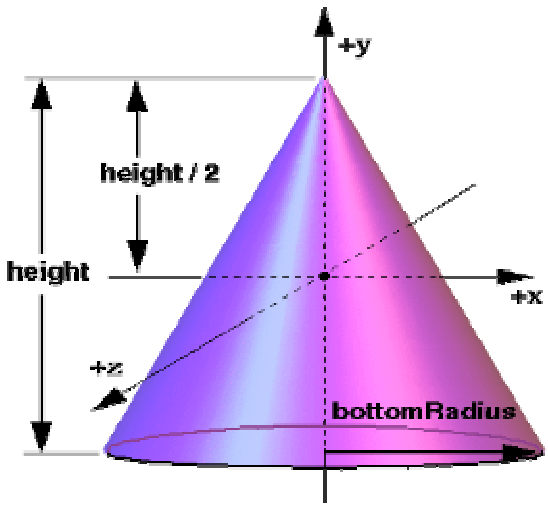



Рисунок 3.2 Параметры, задающие конус

**Cylinder{**

| | | | |
|---|---|---|---|
| field | SFFloat | radius | 1 |
| field | SFFloat | height | 2 |
| field | SFBool | side | TRUE |
| field | SFBool | top | TRUE |
| field | SFBool | bottom | TRUE |

}

Узел типа Cylinder описывает цилиндр, ось вращения которого совпадает с осью Y (см. рис. 3.3).

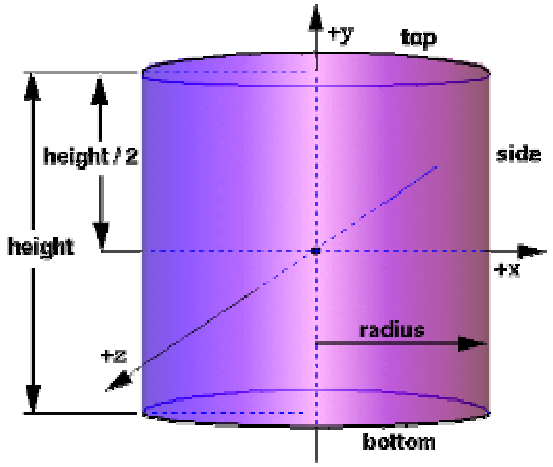



Рисунок 3.3 Параметры, задающие цилиндр

**ElevationGrid{**

| | | | |
|---|---|---|---|
| eventIn | MFFloat | set_height | |
| exposedField | SFNode | color | NULL |
| exposedField | SFNode | normal | NULL |
| exposedField | SFNode | texCoord | NULL |
| field | MFFloat | height | [ ] |
| field | SFBool | ccw | TRUE |
| field | SFBool | colorPerVertex | TRUE |
| field | SFFloat | creaseAngle | 0 |
| field | SFBool | normalPerVertex | TRUE |
| field | SFBool | solid | TRUE |
| field | SFInt32 | xDimension | 0 |
| field | SFFloat | xSpacing | 0.0 |
| field | SFInt32 | zDimension | 0 |
| field | SFFloat | zSpacing | 0.0 |

}

Узел ElevationGrid (сетка высот) используется для описания однородной прямоугольной сетки, прямоугольной сетки высот, определяемых относительно плоскости XZ. Геометрическая форма объекта описывается массивом скалярных величин, каждая из которых представляет собой высоту прямоугольного участка поверхности для каждой точки сетки (см. рис. 3.4).

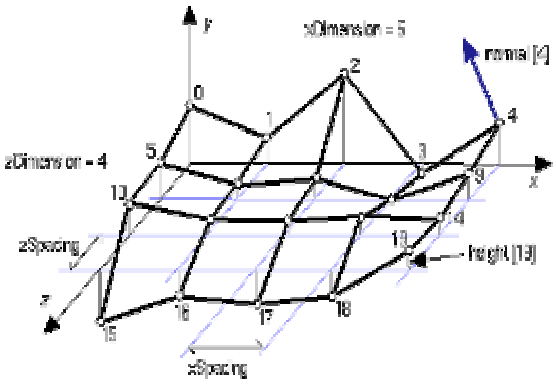



Рисунок 3.4 Параметры, для задания ElevetionGrid.

**Sphere {**

| | | | |
|---|---|---|---|
| field | SFFloat | radius | 1 |

**}**

Узел типа Sphere (сфера) в качестве центра имеет точку с координатами (0, 0, 0). Поле radius узла данного типа содержит числовое значение радиуса сферы и должно быть не меньше 0. Когда на сферу накладывается текстура, то покрывается вся ее поверхность (начиная с обратной стороны) в направлении против часовой стрелки. В точках, где плоскость YZ пересекает сферу, образуется шов текстуры (см. рис. 3.5).

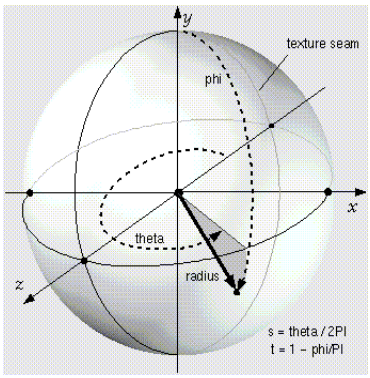



Рисунок 3.5 Параметры сферы

**IndexedFaceSet{**

| | | | |
|---|---|---|---|
| eventIn | MFFloat | set_colorIndex | |
| eventIn | MFFloat | set_coordIndex | |
| eventIn | MFFloat | set_normalIndex | |
| eventIn | MFFloat | set_texcoordIndex | |
| exposedField | SFNode | color | NULL |
| exposedField | SFNode | coord | NULL |
| exposedField | SFNode | normal | NULL |
| exposedField | SFNode | texCoord | NULL |
| field | SFBool | ccw | TRUE |
| field | MFInt32 | colorIndex | [ ] |
| field | SFBool | colorPerVertex | TRUE |
| field | SFBool | corvex | TRUE |
| field | MFInt32 | coordIndex | [ ] |
| field | SFFloat | creaseAngle | 0 |
| field | MFInt32 | normalIndex | [ ] |

| | | | |
|---|---|---|---|
| field | SFBool | normalPerVertex | TRUE |
| field | SFBool | solid | TRUE |
| field | MFInt32 | texCoordIndex | [ ] |

}

Узел IndexedFaceSet (индексированный набор граней) используется для создания многоугольников, вершины которых определяются с помощью списка из поля coord. Этот узел используется при создании объектов сложной геометрической формы.

**Coordinate {**

| | | | |
|---|---|---|---|
| exposedField | MFVec3f | point | [ ] |

}

Узел Coordinate (координаты) позволяет указать набор трехмерных координат, предназначенных для использования в поле coord геометрических узлов, таких как: (IndexedLineSet, IndexedFaceSet и PointSet).

**TextureCoordinate {**

| | | | |
|---|---|---|---|
| exposedField | MFVec2f | point | [ ] |

}

Узел TextureCoordinate содержит двухмерные координаты текстур, “привязываемые” к вершинам грани геометрических узлов типа IndexedFaceSet и ElevationGrid.

**Normal {**

| | | | |
|---|---|---|---|
| exposedField | MFVec3f | vector | [ ] |

}

Узел типа Normal (нормаль) определяет набор векторов нормалей (используемых геометрическими узлами типа ElevationGrid и IndexedFaceSet).

**Color {**

| | | | |
|---|---|---|---|
| exposedField | MFColor | color | [ ] |

}

Узел типа Color (цвет) применяется для указания цвета в формате RGB.

**PointLight {**

| | | | |
|---|---|---|---|
| exposedField | SFFloat | ambientIntensity | 0 |
| exposedField | SFVec3f | attenuation | 1 0 0 |
| exposedField | SFColor | color | 1 1 1 |
| exposedField | SFFloat | intensity | 1 |
| exposedField | SFVec3f | location | 0 0 0 |
| exposedField | SFBool | on | TRUE |
| exposedField | SFFloat | radius | 1 |

}

Это простейший источник освещения, его излучение исходит из одной точки и является всенаправленным. С помощью полей можно задать местоположение, цвет, и интенсивность света.

Список, приведенный выше, является далеко не полным. Остальные узлы, не описанные здесь можно найти в соответствующей литературе [2],[3],[4]. Основной WWW-узел с информацией о VRML, спецификации VRML 2.0, VRML97 находится по адресу: **http://vag.vrml.org/www-wrml/**

# 4. ГИС ArcView как среда разработки

ArcView разработан Институтом Исследований Систем Окружающей Среды (Environmental Systems Research Institute, ESRI), изготовителем ARC/INFO - ведущего программного обеспечения для географических информационных систем (ГИС).

ArcView это мощный, легкий в использовании инструмент для обеспечения доступа к географической информации. ArcView предоставляет широкие возможности для отображения, изучения, выполнения запросов и анализа пространственных данных. Главная особенность ArcView - простота загрузки табличных данных, типа файлов dBASE и данных с серверов баз, для отображения, запросов, обработки и организации таких данных в удобном для восприятия и анализа виде.

Форматы пространственных данных, поддерживаемые ArcView.:

- Шейп-файлы ArcView
- Покрытия ARC/INFO
- GRID данные ARC/INFO
- Растровые данные
- САПР чертежи
- SDE данные (Если установлен модуль доступа к Базам Данных)
- Тип данных «транспортная сеть» (Если установлен модуль StreetMap)
- TINs (Если установлен модуль 3D Analyst)
- VPF данные

Для более полного знакомства с ГИС ArcView обратитесь к соответствующей литературе [5].

## 4.1. Интерфейс пользователя ArcView

В ArcView вы работаете с видами, таблицами, диаграммами и скриптами, сохраненными в одном файле, называемом проектом. Одновременно в ArcView можно работать только с одним проектом. Проект позволяет вам сохранять все компоненты, нужные вам для определенной задачи или приложения.

Когда вы открываете одну из компонент проекта, она отображается в собственном окне. В ArcView вы можете открыть любое количество окон, но одновременно только одно из них будет активным. Активное окно - то окно, с которым вы сейчас работаете.

Когда вы выполняете действия в ArcView, они обычно выполняются над активным окном. Пользовательский интерфейс ArcView изменяется в зависимости от типа активного окна.

## 4.2. Проект (Project)

Проект - это файл, в котором хранится ваша работа, выполняемая с помощью ArcView. Проект обычно содержит все виды, таблицы, диаграммы, компоновки и тексты программ (скрипты), используемые в конкретном приложении ArcView, каждый проект в ArcView имеет окно проекта.

Файл проекта не содержит сами данные, которые используются в ArcView, такие как: пространственные данные типа шейп-файлов (shapefiles) и покрытия ARC/INFO, а также табличные данные типа dBASE файлов. В проекте хранятся только ссылки на местоположение исходных данных на диске.

Проект может включать три типа объектов: документы, графические интерфейсы пользователя (ГИП) и скрипты. Документы предоставляют различные средства визуализации и управления вашими данными: видами, таблицами, компоновками, редакторами скриптов и диаграммами. Графические интерфейсы пользователя (ГИП) определяют средства для управления документами. ГИП могут быть определены системой или быть локальными для проекта. Скрипты создаются на языке Avenue и выполняют в ArcView различные задачи.

Документы обеспечивают средствами для взаимодействия с данными. ArcView поддерживает следующие документы: Views (Виды), Tables (Таблицы), Charts (Диаграммы), Layouts (Компоновки) и Script Editors (Редакторы Скриптов).

В ArcView работа идет с географическими (распределенными в пространстве) данными на интерактивных картах, называемых видами. Каждый вид в ArcView предоставляет уникальную географическую таблицу содержания, облегчающую понимание отображаемых данных. Документ вида предоставляет средства для отображения и запрашивания набора определяемых пользователем тем.

Таблица является одним из документов ArcView. Таблица - это визуальное представление табличных данных. Она использует разграфку, напоминающую электронные таблицы, которые состоят из связанных значений, упорядоченных по горизонтали; и однотипных данных, выстроенных по вертикали. Атрибуты, являются колонками таблицы, а записи образуют строки. Таблицы ArcView обеспечивают полный набор возможностей для получения итоговой статистики, сортировки и запросов.

Диаграмма отображает табличные данные как деловую графику. Каждая диаграмма ассоциируется с таблицей, из которой она выводит данные выбранных записей и выбранных полей. Поля задаются в диалоговом окне свойств диаграммы или из Avenue.

Компоновка - это разновидность документа, которая обеспечивает интерактивные возможности компоновки карты для пользователя ArcView. Каждая компоновка организует список графических объектов. Она затем может рисовать это объекты на экран, посылать их на принтер, и посылать их в один из нескольких стандартных графических файловых форматов. Компоновка - это карта, на которой вы можете показать виды диаграммы, таблицы, импортированную графику и графические примитивы. Компоновка используется для подготовки этих графических объектов для вывода из ArcView.

Скрипт является компонентом проекта ArcView, содержащим код программы на Avenue. Также как макросы, процедуры или скрипты в других языках программирования, скрипты ArcView объединяют средства для выполнения трех общих целей: задачи автоматизирования, добавления новых возможности в ArcView и создания законченных прикладных программ (приложений). Скрипт является основным элементом объектно-

ориентированного языка программирования Avenue. Скрипт Avenue позволяет контролировать способ и время посылки запросов к объектам ArcView. Через Avenue вы можете настроить интерфейс и функциональность ArcView.

## 4.3. Язык Avenue – средство разработки приложений

Avenue является языком программирования и средой разработчика, входящей в состав ArcView. Язык Avenue полностью интегрирован с ArcView, и все, что создано, будет запускаться на любых платформах, которые доступны для ArcView. Можно назвать много целей применения Avenue: можно использовать Avenue для настройки способа работы с ArcView, сделать так, чтобы в ArcView решались необходимые специфичные задачи, или даже разработать законченное приложение, работающее вместе с графическим интерфейсом пользователя ArcView.

ArcView включает необходимые средства настройки и среду языка. Можно создать графический интерфейс пользователя конкретно для любой задачи, установить некоторые начальные свойства графических элементов управления, с которыми будет работать пользователь, а затем настраивать их функционирование и внешний вид, и написать код Avenue, который будет реагировать на состояние элементов интерфейса. Можно привязать скрипты, созданные на Avenue, к другим событиям, таким как открытие проекта или его закрытие.

Avenue является объектно-ориентированным языком программирования. Особенностью Avenue, как и всех объектно-ориентированных языков, является идентификация объектов, посылка к ним запросов. Объект является подобием некоего пакета, заключающего в себе тесно связанные данные и функциональность. В Avenue вместо непосредственного вызова функций с аргументами вы посылаете запрос к объекту. Когда объект получает этот запрос, он выполняет какое-то действие. Объекты ArcView являются элементами иерархии классов, которые организованы по функциональным категориям, связанным со всеми аспектами приложения.

Операторы Avenue используются для организации и структурирования выполнения запроса, то есть они определяют, когда и как он будет выполняться. Запросы Avenue являются развитием традиционного вызова функций. Вызов функций и их выполнение представляют собой однозначное соответствие, тогда как запрос может подключать один из любого числа методов. Запрос задает, какой из элементов данного класса будет задействован, и какой метод при этом будет использоваться. Программирование на Avenue скорее состоит из написания объектных запросов, чем из вызова функций. Посылая запрос к объекту, вы активизируете метод, соответствующий тому классу, элементом которого является данный объект. Объект Avenue всегда отвечает на запрос, возвращая объект, в некоторых случаях запрос создает новый объект, в других случаях исходный запрос возвращает существующий объект.

В Avenue вы устанавливаете и контролируете состояние системы в объектах, созданных как экземпляры классов.

Для более полного знакомства с языком Avenue обратитесь к соответствующей литературе [6].

## 4.4. Динамический обмен данными (Dynamic Data Exchange)

Microsoft поддерживает механизм работы в режиме клиент/сервер, называемый динамическим обменом данными (Dynamic Data Exchange, DDE). DDE делает возможным обмен данными между двумя приложениями. ArcView поддерживает DDE и может связаться с любым другим приложением, поддерживающим технологию DDE, таким как VisualBasic, Excel, Access, Delphi. DDE имеет одно ограничение: приложения, связывающиеся через DDE, должны быть запущены на одном и том же компьютере; DDE не поддерживает работу по сети.

Взаимодействие двух приложений называется диалогом (conversation). Одно приложение является сервером, а другое приложение - клиентом. Диалог DDE определяется двумя параметрами: именем приложения (application name) и темой (topic). Для установки диалога с ArcView, в качестве имени приложения задается "ArcView", а в качестве имени темы - "System". ArcView имеет только один раздел, и он называется system. Тема определяет предмет диалога и обычно является именем, которое определяется по приложению - источнику. Если вы хотите установить диалог с приложением, созданным на Delphi, то в качестве имени приложения задается имя исполняемого файла, а в качестве имени темы - заголовок главной формы. Комбинация имени приложения и темы однозначно определяет диалог и остается постоянной во время всего диалога.

DDE диалог иногда называют связью (link), так как два приложения связываются друг с другом при помощи данных, которыми они обмениваются. DDE поддерживает три различных типа связей: автоматические, связи устанавливаемые вручную, и извещение (уведомление).

- Автоматическая связь (Automatic link) - исходное (source) приложение автоматически обновляет целевое (destination) приложение при каждом изменении данных в исходном приложении.

- Ручная связь (Manual link) - обмен данными не происходит автоматически; целевое приложение должно запросить новые данные у исходного. ArcView поддерживает только этот тип связи.

- Связь-извещение (Notify link) - исходное приложение уведомляет целевое приложение об изменении данных, но целевое приложение должно затем запросить новые данные из исходного.

Имеются три различных способа для работы в диалоге DDE: выполнение (execute), запрос (request) и запись по машинному адресу (poke). При выполнении клиент просит сервер выполнить некоторую функцию. Текстовая строка, которую клиент посылает серверу, обычно является некоторой директивой на языке команд этого сервера.

При запуске приложения ArcView автоматически создается DDEServer. По умолчанию приложение ArcView называется ARCVIEW и сервер DDE поддерживает системную тему. Сервер ArcView поддерживает стандартные транзакции: execute (выполнить), request (запрос) и poke (запись элемента данных).

Execute (выполнить) побуждает сервер выполнить команды, которые он получает в строке от клиента.

Request (запрос) возвращает клиенту запрошенную информацию.

Poke (запись элемента данных) запрашивает подтверждение о передаче данных.

Клиент, подключенный к ArcView серверу, посылает запросы серверу через скрипты Avenue. Для транзакции выполнения клиент передает на сервер строку, являющуюся скриптом на языке Avenue, которая выполняется на сервере.

Транзакция Request (запроса) требует, чтобы клиент назвал элемент, значение которого будет возвращено. Для сервера ArcView элементом служит скрипт Avenue, а возвращаемое значение - это объект возврата этого скрипта, преобразованный в строку.

Транзакция Poke (запись элемента данных) несколько сложнее. Элемент имеет два параметра: данные (которые должны быть строкой) и имя скрипта находящегося на сервере, который будет запущен для выполнения. Данные - это строка любой длины. Данные становятся значением SELF-объекта для скрипта, указанного в первом параметре.

# 5. Построение сцены в VRML

## 5.1. Описание изобразительных средств языка VRML

Язык VRML позволяет создавать очень реалистичные сцены. Это достигается за счет базового набора примитивов, возможности наложения текстур на создаваемые объекты, добавления цветовых и осветительных эффектов. Язык содержит множество примитивов, базовых объектов, на основе которых можно создавать достаточно сложные модели. К базовым геометрическим объектам можно отнести: Sphere, Cone, Cylinder, Box, IndexedFaceSet, ElevationGrid, Extrusion. Каждый из этих примитивов имеет набор свойств, благодаря которым можно существенно менять внешний облик объекта. Кроме того, имеется большое количество узлов, благодаря которым можно достигать очень реалистичных изображений. Эти узлы можно условно разбить на несколько групп.

К первой группе узлов можно отнести узлы отвечающие за освещение сцены. Большинство VRML-браузеров вычисляет освещенность путем определения ближайшей поверхности и интерполяции между вершинами этой поверхности - модель Гуро. У этой модели существуют недостатки. Если поместить источник света над серединой очень большой поверхности мы не получим эффекта блика. Для устранения этого недостатка необходимо большие грани разбивать на более мелкие. Источников освещения в языке VRML существует три:

- Направленный источник освещения (DirectionalLight) – это свет который выходит из бесконечно далекого источника, поэтому все его лучи параллельны. Направленный свет не имеет своего источника в виртуальном мире, только направление. Он является хорошей имитацией солнечного света.

- Точечный источник освещения (PoiintLight) - это свет, исходящий из определенной точки пространства равномерно по всем направлениям. Его аналогом в реальном мире может служить простая лампочка.

- Прожектор (SpotLight) данный узел создает канонический поток света. Этот узел имеет расположение в пространстве, а также направление. Аналогом данного освещения в реальном мире может служить свет от прожектора.

Ко второй группе узлов относятся узлы отвечающие за цвет объектов. Цвета в языке VRML представляются в RGB-формате. Цвет представляет комбинацию красной (Red), зеленой (Green) и синей (Blue) компонент. Значение каждой из компонент должно лежать в диапазоне от 0 до 1. К этой группе узлов можно отнести следующие: Material, Color, Normal, ColorInterpolator, NormalInterpolator. Узлы Material определяют свойства материала и занимают особое место в языке VRML, так как добиться качественного изображения без его использования невозможно. Однако создание реалистичных свойств материалов дело достаточно трудоемкое и сложное. На помощь приходит сеть Internet, где можно найти большое количество библиотек, созданных различными организациями, предоставляющих различные материалы, такие как: серебро, золото, дерево, сталь и др. Использование данных библиотек одно из ключевых моментов для создания качественных, реалистичных 3D-сцен.

К третьей группе узлов относятся узлы, предоставляющие возможность работы с текстурами. Если не использовать технологию наложения текстур, то добавление мельчайших деталей, таких как кнопки, на телефоне-автомате, будет серьезным испытанием для процессора и может сильно снизить скорость передвижения посетителей VRML мира. При помощи наложения текстур можно значительно усилить реализм сцены, не создавая вручную каждую деталь. Однако чрезмерное использование различных текстур в сцене может сильно снизить скорость передачи VRML файла по каналам связи. Поэтому к выбору количества текстур необходимо подходить осторожно. Перечислим основные узлы этой группы, их четыре:

- PixelTexture – используется для задания текстур основанных на двумерных изображениях. Это осуществляется путем явного задания массива значений пикселов и параметров, которые управляют мозаичным размещения повторяющегося элемента текстуры на поверхности геометрического объекта;
- ImageTexture – с помощью данного узла на объекты можно накладывать текстуры, хранящиеся в файлах. Стандарт VRML строго требует поддержки форматов JPEG и PNG, но поддержка браузерами иных форматов не возбраняется;
- MovieTexture – позволяет использовать текстуры, содержащейся в анимационном файле (movie file), а также задать параметры анимации;
- TextureTransform. – определяет двухмерные преобразования координат текстур. С помощью данного узла текстуры можно поворачивать, сдвигать, масштабировать.

К четвертой группе элементом можно отнести узлы отвечающие за специальные эффекты. Использование этих узлов не обязательно, но позволяет повысить общее качество сцены, создать большую реалистичность. К таким узлам можно отнести:

- Background – определяет фон изображения;
- Fog – туман, с помощью данного узла можно создавать эффекты, имитирующие различные атмосферные явления. Вне границ видимости никакие объекты видны не будут;
- Sound – отвечает за звуковые эффекты. Использование звуковых эффектов позволяет «оживить» сцену;

## 5.2. Сопоставление картографических и VRML объектов

VRML разрабатывался как средство передачи и отображения взаимосвязанных объектов и явлений реального мира. К сожалению, современная вычислительная техника пока не позволяет отображать виртуальное пространство с кинематографической степенью соответствия действительности и в режиме реального времени. Следствием этого является разочарование, которое испытывает большинство людей при первом знакомстве с VRML. Однако, наибольший интерес в VRML вызывает совсем не то, что было до сих пор создано, а то, что заложено и может быть реализовано в будущем.

VRML – это язык на котором трудно программировать «вручную». Одни только математические вычисления в значительной степени затрудняют разработку 3D миров.

Создание геометрических объектов имеющих сложную форму очень сложный и трудоемкий процесс.

При создании моделей реального мира часть работы возлагается на зрителя. Удивительно, насколько легко человеческий мозг может создать целостный визуальный образ сцены и дополнить ее воображаемыми подробностями, даже если они не существуют в действительности. Благодаря этой возможности человеческого мозга задача сопоставления картографических знаков VRML-объектам несколько облегчается.

Сопоставление картографических условных знаков и их трехмерных представлений:

- Внемасштабные условные знаки (точечные) - для каждого такого условного знака может быть создан соответствующий ему 3D-объект, который впоследствии будет просто встраиваться в виртуальный мир (например, дерево, столб, памятник, светофор и т.п.). Управляя цветом и размером можно отображать различные атрибутивные свойства картографических объектов.

- Линейные условные знаки (линейные) - 3D-объекты на основе линейных условных знаков должны создаваться динамически в зависимости от геометрической формы картографических условных знаков (например, такие объекты реального мира как реки в виртуальном мире могут быть представлены в виде набора граней, на которые может быть наложена текстура «водной поверхности»). Для линейных объектов также важны такие характеристики как высота (ограждения), высота над поверхностью (линии электропередачи), линии на плоскости, имеющие некоторую ширину (линейные гидрографические объекты), сложно-графические линии (рельсы).

- Площадные условные знаки (полигональные) - 3D-объекты на основе площадных условных знаков должны создаваться динамически в зависимости от геометрической формы картографических условных знаков (например, такие объекты реального мира как здания и сооружения в виртуальном мире могут быть представлены в виде многоугольного параллелепипеда). Площадные знаки могут быть расклассифицированы на техногенные объекты, имеющие высоту, высоту над поверхностью, граница этих объектов показывает грани, и площадные ареалы. Последние необходимо показывать, заполненными некоторыми символами, которые позволили ли бы распознавать, что это за ареал – лес, болото или территория с заданной плотностью распространения некоторого явления.

## 5.3. Построение 3D сцены в ArcView

Для построения 3D сцены автором был предложен следующий механизм:

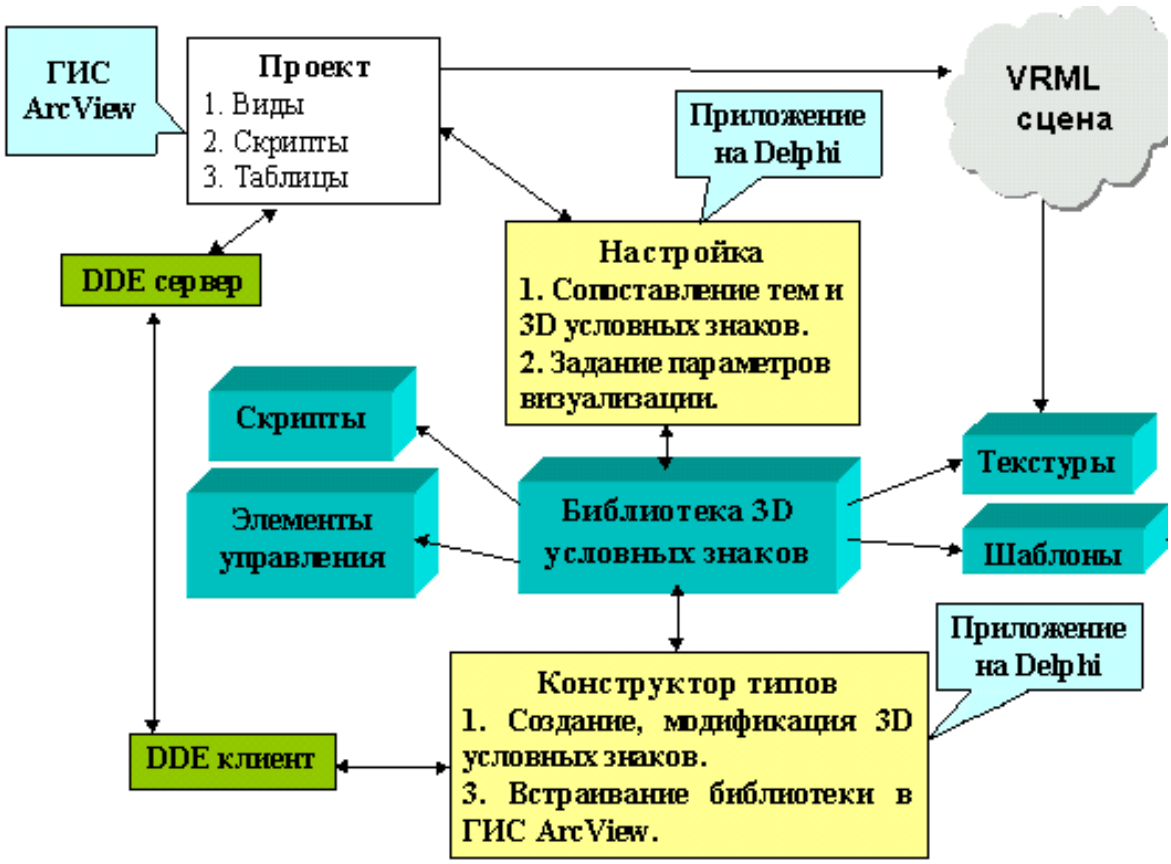



Рисунок 5.1. Предложенный механизм генерации 3D модели.

Для получения 3D модели городской территории необходимо:

1. Инсталлировать библиотеку в проект;
2. Сопоставить темы вида(View) и 3D условные знаки, настроить параметры генерируемых 3D знаков;
3. Настроить параметры генерируемой модели местности;
4. Создать выборку 2D объектов;
5. Активизировать инструмент «сгенерировать модель» для запуска процесса построения модели на основе выбранных объектов;
6. Отобразить сгенерированную сцену в браузере.

Генерация описания осуществляется следующим образом. Каждой теме ставиться в соответствие «трехмерный» условный знак, это соответствие определяет, как в VRML-сцене будут выглядеть объекты данной темы. Для каждого объекта темы выполняется скрипт Avenue, соответствующий «трехмерному» условному знаку. Этот скрипт генерирует строку, являющуюся правильной конструкцией на языке VRML(см. рис. 5.1.).

## 5.4. Моделирование поверхности

ArcView поддерживает различные форматы для работы с поверхностями:

- Формат для представления поверхности на основе нерегулярной сети треугольников (модуль ARC/INFO TIN);
- Формат для представления поверхности на основе регулярной матрицы (модуль ARC/INFO GRID)

Оба данных формата являются закрытыми, в среде ArcView есть специальные модули (Spatial Analyst и 3D Analyst), которые позволяют воспроизводить поверхности, но получить информацию о модели интересующей территории в программе нельзя (можно только использовать стандартные функции).

В данной работе в качестве модели данных представляющей поверхность была выбрана триангуляционная нерегулярная сеть (TIN)[8],[9]. Данные хранятся в текстовом файле, имеющем следующую структуру:

| | |
|---|---|
| “TIN FILE” | - заголовок файла; |
| “Point Count” | - строка; |
| <число> | - количество точек; |
| “Triangle Count” | - строка; |
| <число> | - количество треугольников; |
| “Points” | - строка; |
| <три числа> | - точка x y z; |
| …….. | |
| “Triangles” | - строка; |
| <шесть чисел> | - три индекса точек, три индекса треугольников. |
| …….. | |

Файл триангуляции может иметь заголовочный файл. В заголовочном файле хранятся настройки визуализации поверхности. Настройка происходит следующим образом:

1. Находится диапазон высот, ищется min и max высоты в триангуляции.
2. Этот диапазон разбивается на несколько уровней.
3. Для каждого уровня можно определить цвет, который будут иметь вершины данной высоты.

Заголовочный файл, текстовый файл имеющий следующую структуру:

| | |
|---|---|
| “Header file from TIN” | - заголовок файла; |
| <число> | - количество точек; |
| <число> | - количество треугольников; |
| <число> | - min высота в триангуляции; |
| <число> | - max высота в триангуляции; |
| <число> | - количество уровней; |
| <число> | - min высота уровня; |
| <число> | - max высота уровня; |
| <число> | - цвет высоты в данного уровня; |
| …… | - три числа, зависит от количества уровней. |

Более полное описание настройки параметров визуализации поверхности смотрите в руководстве пользователя.

Модель поверхности в языке VRML описывается узлом IndexedFaceSet (индексированный набор граней). Если используется заголовочный файл триангуляции, то для определения цвета вершины в диапазоне соответствующего уровня значение интерполируется.

При добавлении модели поверхности, сложной задачей становиться расположение объектов на поверхности.

В данной работе использовались следующие методы, применяемые при расположении объектов на плоскости:

*Точечные объекты.* Обычно 3D аналоги таких объектов не обладают большой площадью соприкосновения с поверхностью. Для данных объектов находим треугольник, в который попадает данная точка (проводится локализация точки). Высота объекта над поверхностью вычисляется на основе барицентрических координат. С помощью данного метода может вычисляться высота для таких точечных объектов как: деревья, столбы, светофоры (см. рис. 5.2). Если же площадь 3D аналога объекта велика, то определять высоту объекта над поверхностью можно как для полигональных объектов.

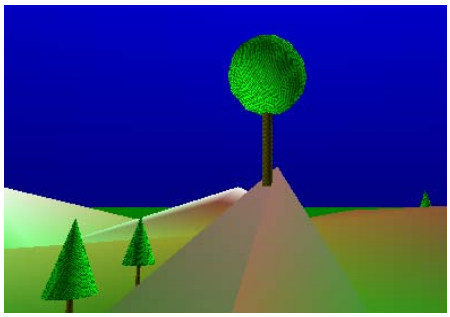


Рисунок 5.2. Расположение на поверхности точечного объекта.

*Полигональные объекты.* 3D аналоги таких объектов обладают значительной площадью соприкосновения с поверхностью. Используется два метода для вычисления высоты над поверхностью для полигональных объектов. Данные объекты состоят из набора точек. Проводится локализация всех точек объекта, на основе барицентрических координат для каждой точки вычисляется высота. В первом методе за высоту объекта над поверхностью берется минимум из высот. Во втором методе, при создании объекта, учитываются все высоты. Первый метод применяется для объектов «здания». При таком методе вычисления высоты над поверхностью возможны ситуации, когда объект будет «провисать» либо «закапываться в землю» (см. рис.5.3).

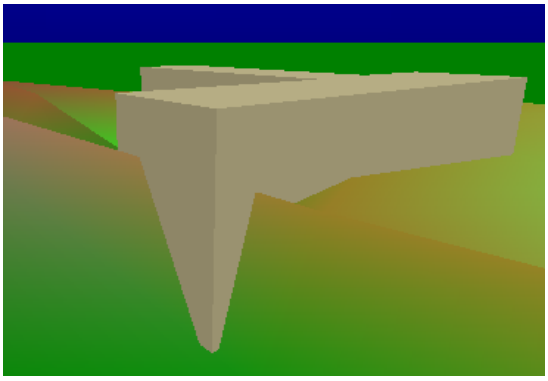
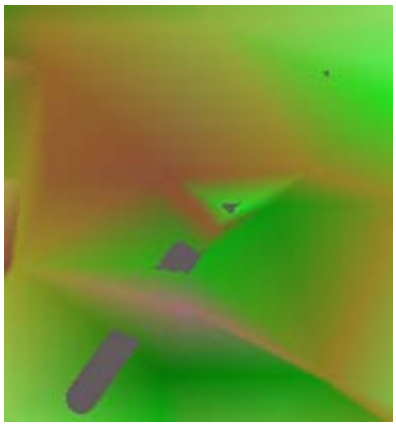


Рисунок 5.3. Здание «закопанное в землю»

Данные недостатки для некоторых типов объектов допустимы, поскольку позволяют сохранить общее представление об объекте.

*Линейные объекты.* Такие объекты представляют собой полилинию. Обычно, 3D аналоги таких объектов имеют небольшую высоту (например, дороги, реки и т.п.). Важность же таких объектов велика, «закапывать в землю» такие объекты нежелательно (см. рис 4.3.). Поэтому для правильной визуализации таких объектов необходимо добавить в полилинию все точки пересечения данной полилинии с ребрами триангуляции. Далее создание 3D-объекта можно осуществлять как для точечных либо полигональных объектов.

Рисунок 5.4. Дорога «закопанная в землю».



К созданию 3D-объектов с учетом поверхности необходимо подходить, учитывая тип объекта, важность объекта, геометрическую форму. Поэтому метод для определения высоты над поверхностью необходимо выбирать для каждого типа объектов в отдельности.

При генерации моделей городских территорий наиболее важными объектами можно считать здания. В данной работе создание таких объектов производилось на основе полигона. Необходимо заметить, что в городах очень часто здания строились по типовым проектам. Поэтому создание типовых моделей зданий, с возможным наложением текстур, могло бы существенно улучшить внешний вид генерируемых сцен.

В заключении можно отметить, что создание конверторов преобразующих стандартные файловые форматы, обеспечивающие работу с поверхностями, в используемый в данном проекте файловый формат, позволило бы использовать накопленные к этому времени реальные данные.

# Заключение

В рамках данной дипломной работы была реализована возможность создания трехмерной модели местности в ГИС ArcView на основе языка VRML. Предложен механизм, который каждому тематическому слою позволяет сопоставить 3D объект, выступающий в роли условного знака при визуализации и задать параметры его генерации, а также произвести визуализацию объектов совместно с визуализацией поверхности. В работе разработаны средства, позволяющие разрабатывать библиотеку трехмерных объектов, а также встраивать эту библиотеку в любой проект ArcView. В связи с тем, что интерес к VRML в настоящее время велик, в сети Интернет существует уже множество описанных на VRML объектов, в ходе дальнейшей работы можно собрать большую библиотеку для получения реалистичных изображений. Сейчас библиотека включает 20 объектов. Реализованный подход позволяет получать модели местности, динамично задавая набор объектов и территориальный охват. В ходе дальнейшей работы можно реализовать возможность подписи объектов в сцене на основе атрибутов, а также задания маршрутов движения по VRML-миру.

## Список использованных источников


1. Салищев К.А. Картоведение.- М.:Издательство московского университета, 1976-438с.
2. Титтел Э. П. и др. Создание VRML-миров. -СПб.: БХВ-СанктПетербург, 1997 -319с.
3. Аврамова О. Д. Язык VRML практическое руководство.- М.: Диалог-Мифи, 2000 – 288с.
4. The Virtual Reality Modeling Language Specification – version 2.0. August,1996.
5. Arcview GIS.ESRI Inc.-NY, 1997г.-376 стр.
6. Avenue. Customization and application development for ArcView. GIS.ESRI Inc.-NY, 1997г.-280 стр.
7. Демерс М.Н. Географические информационные системы основы. - М.: Дата+, 1999 – 493с.
8. Препарата Ф., Шеймос М. Вычислительная геометрия: Введение / Пер. с англ - М.: Машиностроение, 1980г. – 204с.
9. Скворцов А.В. Триангуляция Делоне и её применение. - Томск: Из-во Том. ун-та, 2002г. – 128с.

# Приложение А. Руководство пользователя

Здесь описаны различные этапы работы с данной библиотекой:

Во-первых, необходимо установить модуль предоставляющий возможность изучать VRML миры. Автор использовал модуль Cortona, созданный ParallelGraphics. Этот модуль «надстройка» над браузерами InternetExplorer или NetscapeNavigator. Поскольку форматы VRML1.0 и VRML2.0 не совместимы, а в данной работе использовались оба, также необходимо установить конвертер, например, созданный ParallelGraphics, Cortona VRML1.0 Converter.

Далее необходимо добавить в проект AcrView возможность создания 3D моделей городских территорий. Для этого необходимо открыть в ArcView тот проект, с которым вы хотите дальше работать. Запустите «Конструктор» типов, для этого необходимо запустить файл prCreator.

**Замечание:** *так как ArcView не умеет правильно обрабатывать символы кириллицы, путь до файла должен состоять только из символов латинского алфавита.*

Выберите в меню пункт ArcView (см. рис А.1.) подпункт DDE Server. Откроется диалог для настройки DDE сервера, смотрите раздел «Настройка параметров DDE сервера». Настроив DDE сервер, выберите пункт меню ArcView подпункт Install. В открытый проект ArcView будут добавлены все средства для использования данной библиотеки. Использование «трехмерной» библиотеки смотрите в разделе «Использование библиотеки в ГИС ArcView».

**Замечание:** *если инсталляция прошла успешно, то, открыв любой вид (View), вы должны увидеть пункт меню «VRML», а также две дополнительные кнопки на панели инструментов.*

Если вы после инсталляции перенесли файл проекта ArcView в другую директорию либо добавили новые «трехмерные» условные знаки, вы должны будете переустановить библиотеку в данный проект.

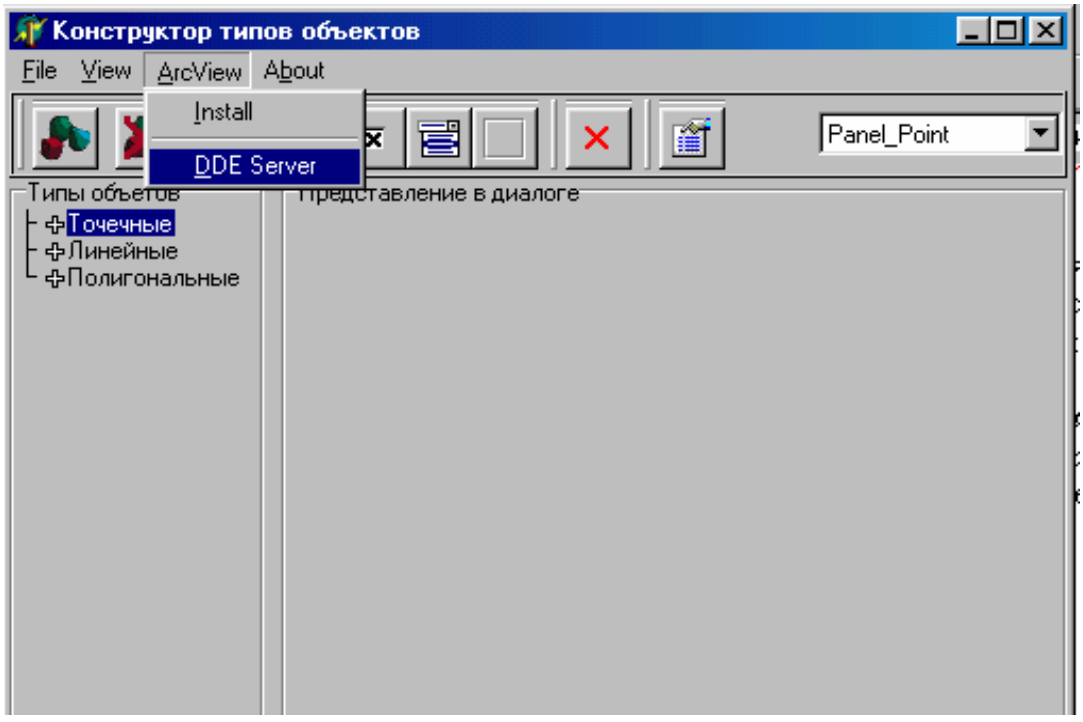



Рисунок А.1. «Конструктор» типов, пункт меню ArcView.

### Настройка параметров DDE сервера

При запуске ГИС ArcView автоматически создается DDE сервер позволяющий передавать данные в ArcView из других приложений. В начале работы с «Конструктором» необходимо настроить параметры DDE сервера. Более подробную информацию о передаче данных между приложениями на основе DDE смотрите в соответствующем разделе, либо в справочной системе ArcView.

Для настройки параметров DDE сервера используется диалог (см. рис. А.2.).

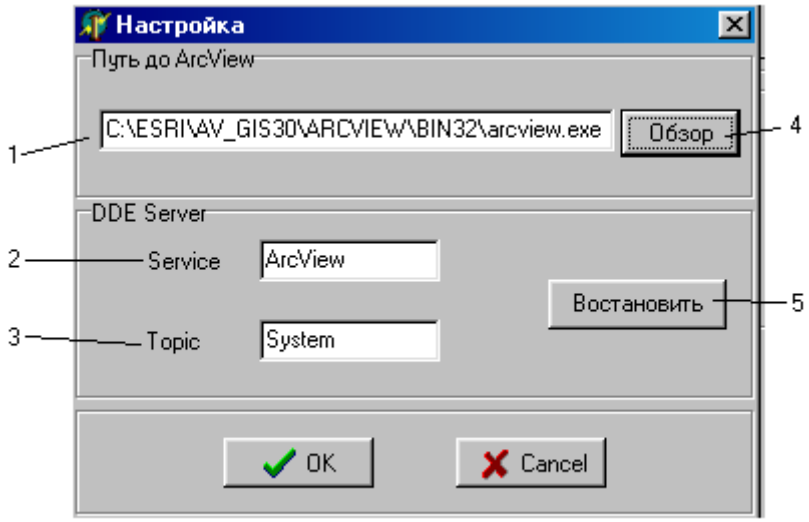



Рисунок А.2. Настройка параметров DDE сервера.

**Содержимое диалога (см. рис. А.2.):**

1. Путь до приложения ArcView. Здесь необходимо указать полный путь до файла arcview.exe. Если во время работы в «конструкторе» приложение AcrView не запущено то «конструктор» запустит его автоматически.
2. Свойство service DDE сервера.
3. Свойство topic DDE сервера. При запуске ArcView по умолчанию значение свойств Sevice = «ArcView», Topic = «System». Изменять эти свойства в диалоге, нужно только в случае, когда они после запуска ArcView были изменены программно.
4. Кнопка «Обзор» открывает стандартный диалог открыть файл (Open file), для указания местоположения ArcView.
5. Кнопка «Восстановить» задает свойства Service = «ArcView», Topic = «System».

### Использование библиотеки в ГИС ArcView

После инсталляции в проект у видов(View) добавляется дополнительный пункт меню, а также две кнопки на панели инструментов (см. рис. А.3.).

Перед генерацией VRML мира необходимо указать, какими 3D объектами представлять тему. Для этого можно выбрать пункт меню «Настроить темы» либо нажать правую кнопку на панели инструментов «2» (см. рис. А.3.). Откроется диалог «Настройка», который позволяет сопоставить темы и «трехмерные» условные знаки, указать путь до файла триангуляции, а также задать параметры визуализации. Описание работы с диалогом «Настройка» смотрите в разделе «Настройка параметров визуализации

тем». После того как вы произвели настройку тем, вы можете генерировать VRML миры. Для этого необходимо выполнить следующие шаги:

- Активизировать темы, объекты которых должны быть в VRML сцене. Активизировать несколько тем, можно щелкая на имени темы мышью удерживая при этом клавишу «Shift». Активные темы в таблице содержания выделены обрамляющим прямоугольником.
- Далее необходимо сделать выборку объектов. Для этого можно воспользоваться инструментом «3» (см. рис.А.3.). Также выборку можно построить с помощью запроса. Построение запросов смотрите в справочной системе ArcView. Выбранные объекты помечаются специальным цветом, по умолчанию желтым цветом.

**Замечание:** *выбирать объекты можно только из активных тем*;

- Выбрав пункт меню «Генерировать модель» или нажав левую кнопку на панели инструментов будет сгенерирован файл VRML. Если в диалоге «Настройка» указан путь до браузера, то автоматически этот браузер будет открыт и вы сможете исследовать только что сгенерированный VRML мир.

Если работа в проекте ArcView с библиотекой закончена, выберите пункт меню «Uninstall» и из проекта будут удалены все встроенные возможности.

**Замечание:** *при де инсталляции VRML файлы не удаляются автоматически, поэтому вам необходимо это делать вручную.*

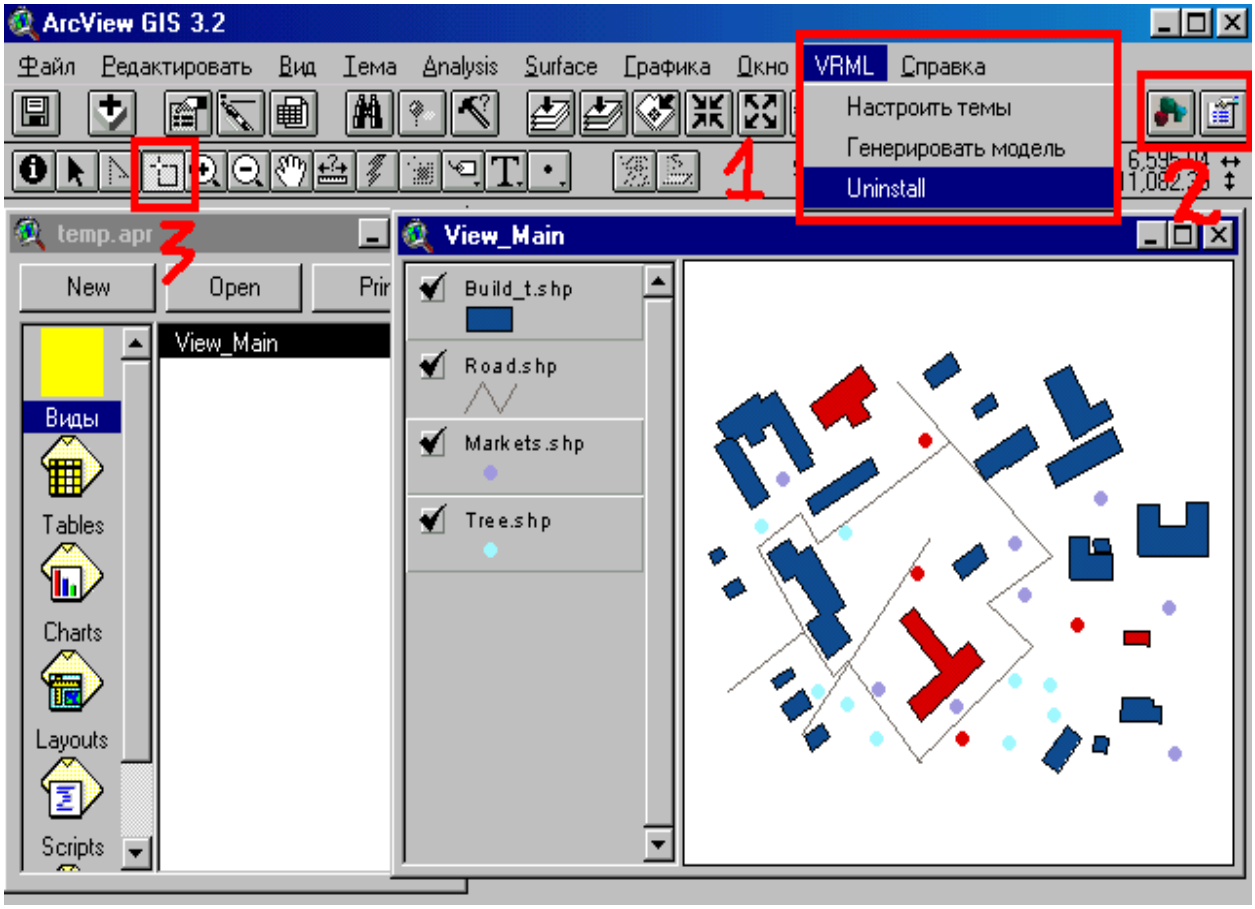



Рисунок А.3. Вид ArcView.

### Настройка параметров визуализации объектов тем

Для настройки параметров визуализации объектов тем существует диалог. В этом диалоге вы можете настроить общие параметры визуализации, а также сопоставить темы с «трехмерными» условными знаками. Данный диалог содержит две страницы «Общие» (см. рис.А.4) и «Темы» (см. рис.А.5). На странице «Общие» вы настраиваете свойства для файла в целом, на странице «Темы» вы сопоставляете темы с «трехмерными» условными знаками, а также задаете параметры визуализации.

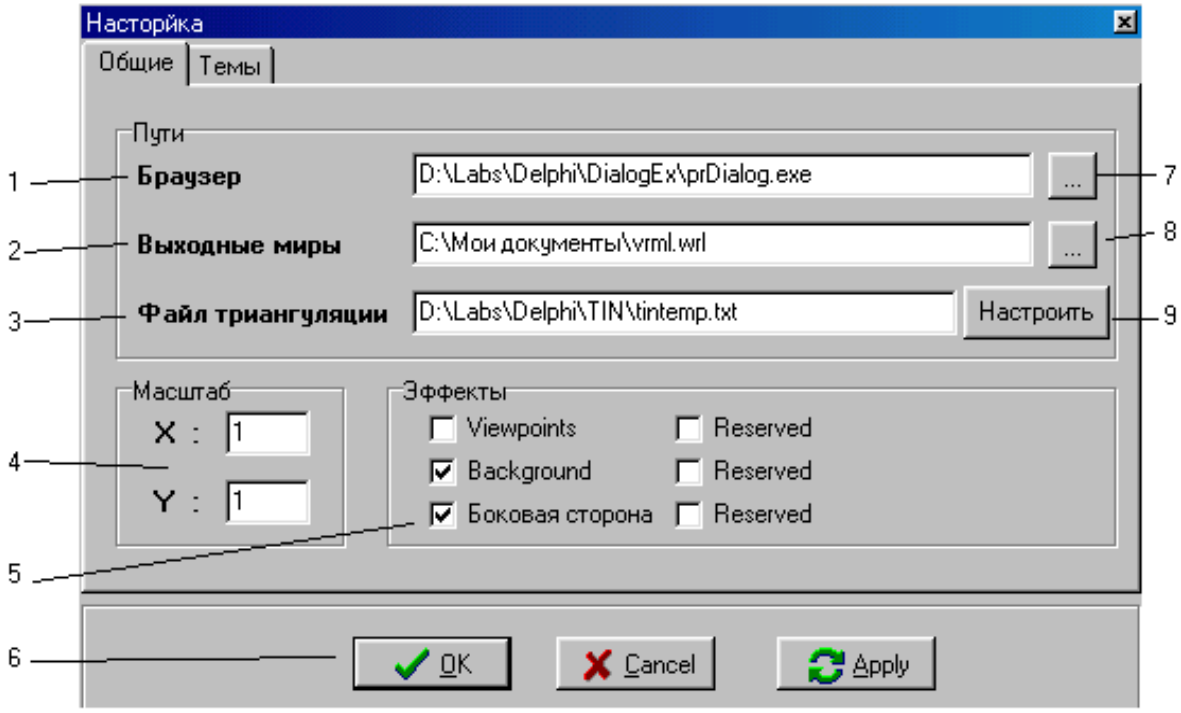



Рисунок А.4. Диалог «Настройка» страница «Общие».

**Содержимое диалога «Настройка» страница «Общие» (см. рис. А.4.)::**

1. **Браузер.** Здесь необходимо указать путь до браузера, который «понимает» формат VRML. Указав данный параметр, после генерации VRML сцены, будет автоматически запущен указанный браузер и ему в качестве параметра будет передан сгенерированный файл. Для просмотра VRML миров необходимо, чтобы на компьютере была установлена программа поддерживающая формат VRML. Существуют специальные модули для Internet Explorer или Netscape Navigator позволяющие просматривать VRML миры в этих браузерах;
2. **Выходные файлы**. Здесь необходимо указать путь и имя файла, в который будут генерироваться VRML миры. В данной работе не предусмотрено удаление сгенерированных VRML файлов, удаление ненужных файлов возложено на пользователя.

   **Совет:** *желательно завести отдельный каталог, в который будут складываться все сгенерированные файлы;*

3. **Файл триангуляции.** Здесь указывается путь до файла триангуляции. Указывайте этот параметр, если вы хотите работать с моделью поверхности;
4. **Масштаб.** Здесь указываются масштабные коэффициенты. Используйте эти коэффициенты, если вы хотите изменить размеры сгенерированных объектов. Например, вы хотите увеличить размер деталей, задав параметры X=0.5 и Y=0.5 вы увеличите размеры объектов в два раза;
5. **Эффекты.** Здесь вынесены специальные эффекты, добавление которых не обязательно, но их использование улучшает качество визуализации:
   - Viewpoints – добавляет в выходные файлы пять точек наблюдения;
   - Background – добавляет в выходные файлы фон «неба и земли»;
   - Боковая сторона – если используется файл триангуляции, то добавляется боковая сторона;
6. **Стандартные кнопки диалога:**
   - Ok – сохранить параметры настройки и закрыть диалоговое окно;
   - Cancel – закрыть диалоговое окно не сохраняя параметры настройки;
   - Apply – сохранить параметры настройки.
7. **Кнопка.** Открывает стандартный диалог (Open file) для указания пути до браузера;
8. **Кнопка.** Открывает стандартный диалог (Open file) для указания пути до выходных файлов;
9. **Кнопка «Настроить».** Открывает диалог для настройки работы с файлом триангуляции. Работа с этим диалогом описана в разделе *«Настройка файла триангуляции»*.

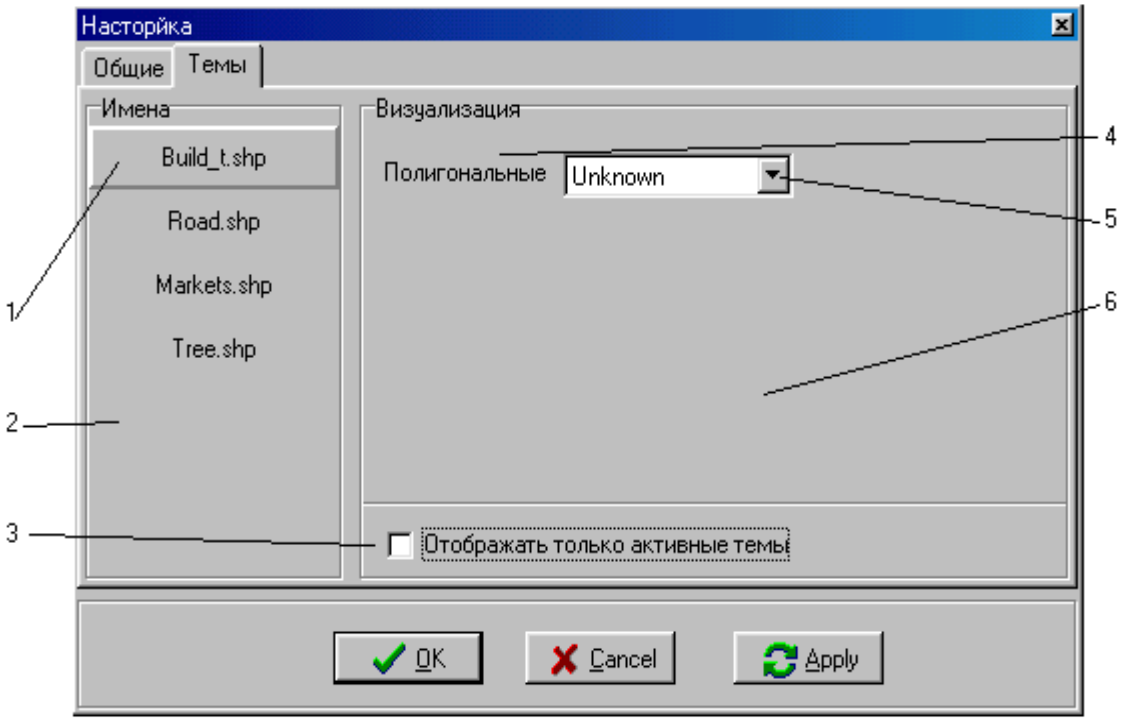



Рисунок А.5. Диалог «Настройка» страница «Темы».

Сопоставление тем и условных знаков происходит на странице «Темы». Сопоставление должно быть проделано для всех тем, объекты которых вы хотите видеть в сгенерированных VRML мирах. Сопоставление происходит для каждой темы в

отдельности, т.е. вы выбираете тему и ставите ей в соответствие «трехмерный» условный знак и настраиваете параметры визуализации (если они есть).

**Содержимое диалога «Настройка» страница «Темы» (см. рис. А.5.):**

1. **Выделенная тема.** Вокруг выделенной темы появляется обрамляющий прямоугольник, в правой части окна вы видите соответствующие выбранной теме параметры визуализации;
2. **Имена.** Здесь вы можете переключать выделенные темы. Для переключения на другую тему необходимо щелкнуть мышью на имени темы. При переключении меняется также параметры визуализации, отображаемые с правой стороны окна, теперь они будут соответствовать новой выделенной теме;
3. **Отображать только активные темы.** Используя этот флаг вы можете сделать невидимыми неактивные в данный момент темы. Настраивать нужно только те темы объекты, которых должны присутствовать в VRML сцене, остальные темы настраивать не обязательно;
4. **Размерность.** Каждая тема в ArcView имеем размерность, т. е. все объекты данной темы имеют такую размерность. Используется три типа размерности:
   - 0 – точечные объекты;
   - 1 – линейные объекты, обладают длиной;
   - 2 – полигональные объекты, обладают площадью.
5. **Выпадающий список.** Содержит список «трехмерных» условных знаков. В зависимости от размерности изменяются содержимое списка. Например, линейные объекты могут быть представлены как дороги, площадные объекты могут быть представлены как здания и т.д.;
6. **Параметры визуализации.** Выводимые здесь элементы управления зависят того какой выбран в выпадающем списке «5» (см. рис. А.5.) «трехмерный» условный знак. Здесь вы задаете параметры, которые имеет этот условный знак (см. рис. А.6).

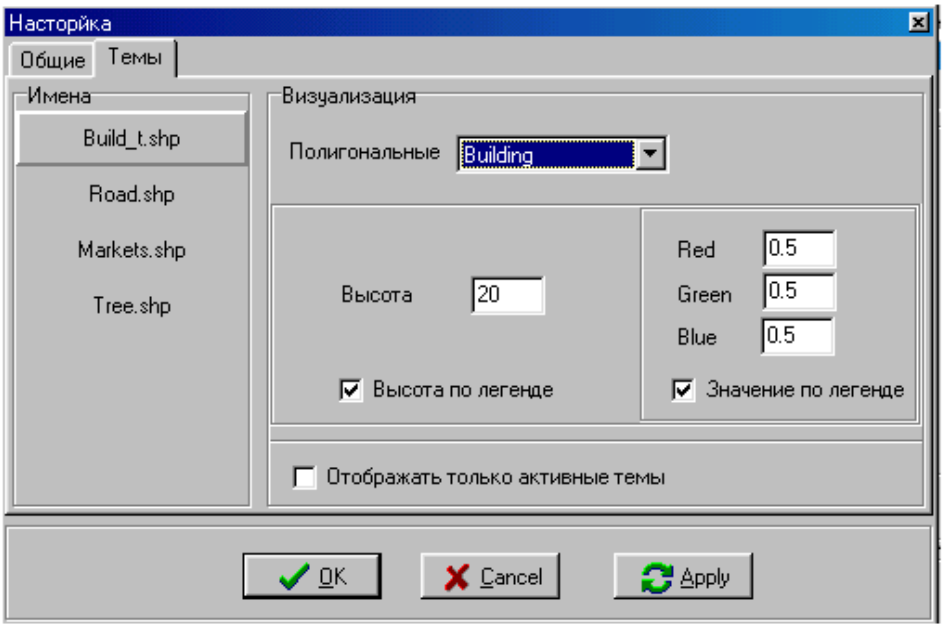



Рисунок А.6. Параметры визуализации «трехмерного» условного знака “Building”.

### Настройка файла триангуляции

Для генерации модели местности необходимо указать файл, содержащий данные, а также настроить параметры визуализации. Это можно сделать в специальном диалоге (см. рис. А.7).

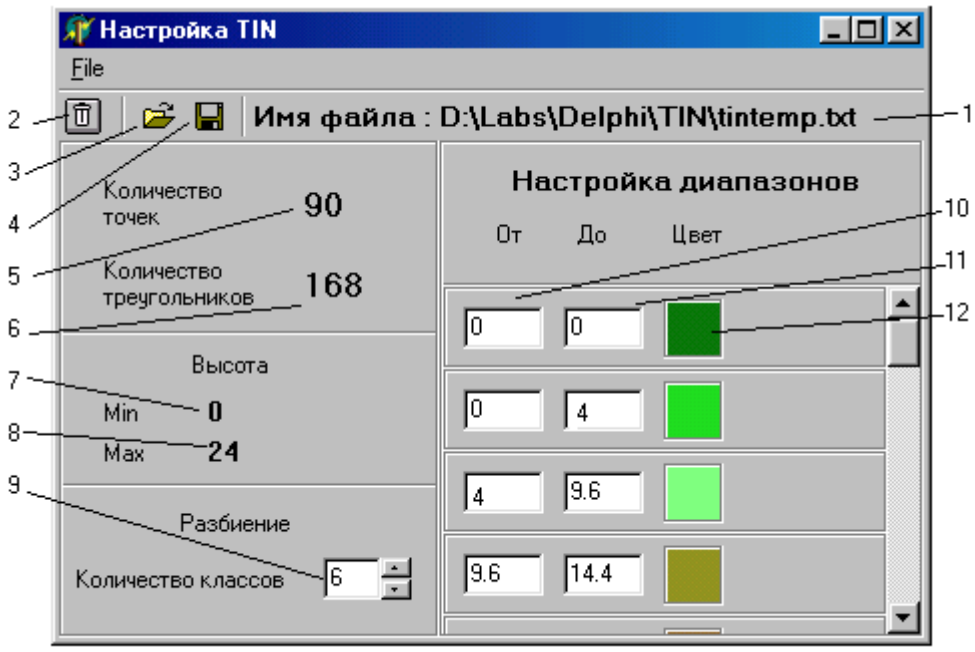



Рисунок А.7. Настройка визуализации триангуляции.

**Содержимое диалога «Настройка TIN» (см. рис. А.7.):**

1. Полный путь до файла триангуляции.
2. Кнопка для сброса всех настроек. Позволяет отказаться от использования модели поверхности в VRML мире;
3. Кнопка открывает диалог (Open file), для выбора файла триангуляции;
4. Кнопка сохраняет параметры визуализации;
5. Количество точек в файле триангуляции;
6. Количество треугольников в файле триангуляции;
7. Минимальное значение высоты (Z-координаты) в файле триангуляции;
8. Максимальное значение высоты (Z-координаты) в файле триангуляции;
9. Количество уровней, на их основе будет вычисляться цвет вершины в VRML модели поверхности. Каждый уровень имеет диапазон (начало и конец) и цвет для вершин высотой равной концу диапазона. Конец диапазона уровня совпадает с началом диапазона следующего уровня;
10. Начало диапазона;
11. Конец диапазона;
12. Кнопка «Цвет». Открывает стандартный диалог для выбора цвета (Color Dialog). Позволяет выбрать цвет для высот данного уровня. Цвет определенный здесь будут иметь только вершины высотой точно равной концу диапазона.

**Как вычисляется цвет вершины:**

- Находим уровень, в который попала высота (Z-координата) вершины. Значение высоты лежит в диапазоне данного уровня;
- Цвет будет нечто среднее между цветами этого и предыдущего уровня. Чем ближе значение высоты вершины к концу одного из диапазонов уровней, тем цвет вершины будет более похож на цвет этого уровня (интерполяция), т.е. если высота совпадает с концом диапазона, то цвет вершины будет совпадать с цветом определенным для данного уровня.

*Цвет уровня определяется для вершин высотой равной концу диапазона.* Минимальное количество уровней два. Первый уровень с диапазоном от min до min высоты. Этот уровень определяет цвет вершин с минимальной высотой и не может быть отредактирован. Определение цветовой шкалы для диапазонов позволяет добиться большей наглядности при визуализации поверхности (система изолиний).

## Создание новых «трехмерных» условных знаков

Данный раздел будет посвящен созданию новых и модификации уже имеющихся «трехмерных» условных знаков.

**Замечание:** *после изменения библиотеки вам необходимо будет переустановить в проекты ArcView данную библиотеку*.

Для начала работы вам необходимо запустить файл «prCreator.exe». Настроить DDE сервер, выбрав в меню пункт DDE Server, подробнее о настройке DDE сервера смотрите раздел «Настройка параметров DDE сервера». После этого вы можете приступать к работе. «Конструктор» типов имеет два режима работы «Dialog»(см.рис.А.8) и «Script»(см. рис. А.9).

Создание нового типа включает в себя два этапа:

- определение элементов управления. Эти элементы управления пользователь увидит в диалоге «Настройка», когда сопоставит теме этот «трехмерный» условный знак. Добавляйте элементы управления, если вы хотите, чтобы пользователь имел возможность настраивать параметры визуализации «трехмерного» условного знака. Добавление новых элементов управления и модификацию существующих вы можете производить в режиме «Dialog». Этот этап не обязателен. Подробнее об этом этапе смотрите в разделе «Создание и модификация элементов управления»;
- создание скрипта на языке Avenue. Этот этап реализуется в режиме «Script». Здесь вы должны реализовать на языке Avenue скрипт, который должен обязательно возвращать строку, соответствующую правильному описанию 3D модели на языке VRML. Этот этап является обязательным. Подробнее об этом этапе смотрите в разделе «Создание, модификация скрипта».

Переключение между режимами осуществляется через пункт меню «View».

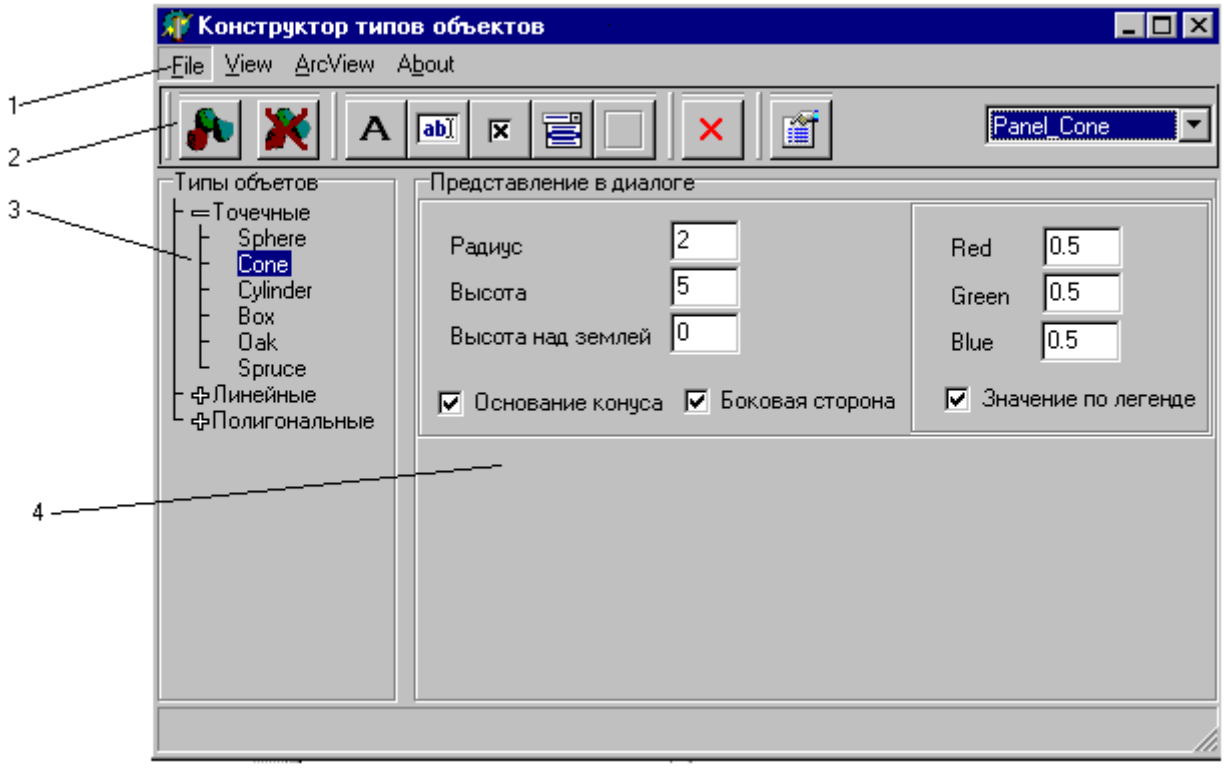



Рисунок А.8. Конструктор типов в режиме «Dialog».

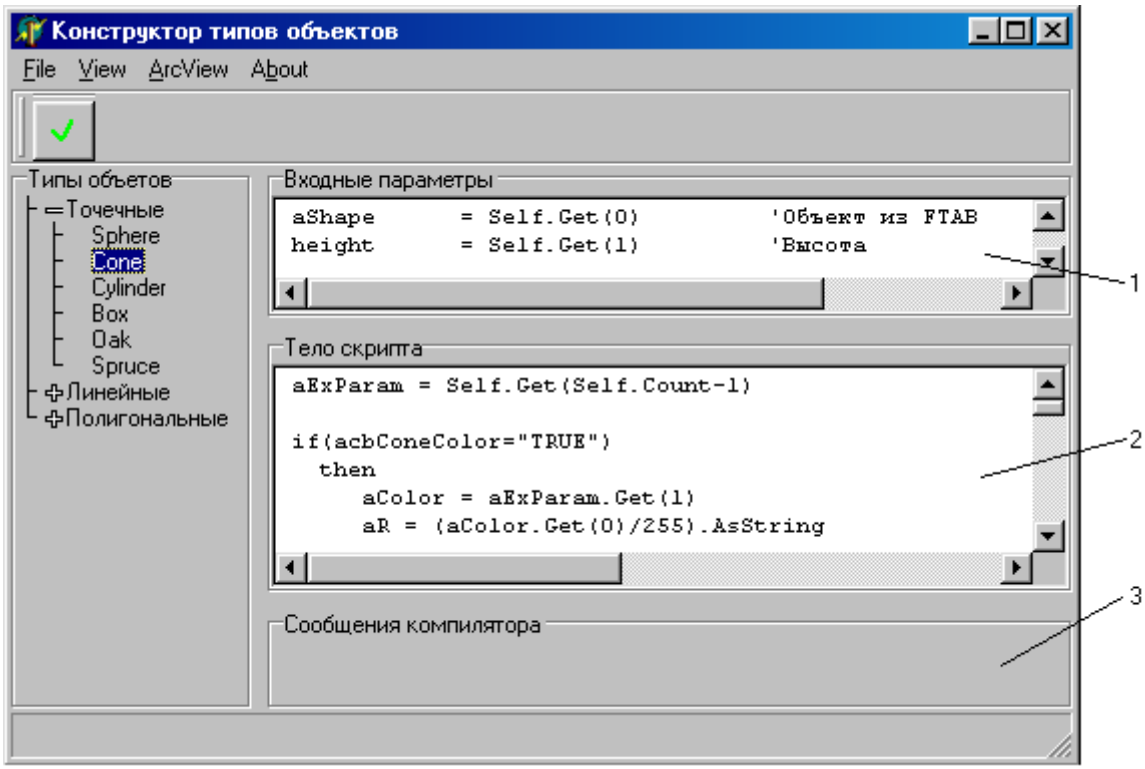



Рисунок А.9. Конструктор типов в режиме «Script».

**Содержимое «Конструктора» в режиме «Dialog» (см. рис. А.8.):**

1. **Главное меню**. Содержит следующие пункты:
   *1.1.* ***File***;
      1.1.1. **Save** – сохраняет изменения для активного «условного» знака.
      1.1.2. **Save All** – сохраняет все внесенные изменения.
   1.2. ***View*** – переключение между режимами;
      1.2.1. **Dialog** - переключение в режимам «Dialog».
      1.2.2. **Script** - переключение в режимам «Script». Данный режим доступен, если на компьютере установлен ArcView и настроен DDE сервер
   1.3. ***ArcView***;
      1.3.1. **Install** – инсталляция библиотеки в проект ArcView.
      1.3.2. **DDE Server** – настройка параметров DDE сервера.
2. **Панель инструментов.** В зависимости от текущего режима меняется содержимое панели инструментов.
3. **Список «трехмерных» условных знаков.** Здесь представлены все условные знаки присутствующие сейчас в библиотеке. Они разделены на три группы: точечные, линейные, полигональные. Все манипуляции вы производите с активным в данный момент условным знаком. Активный знак находится в синей рамке.
4. **Область действий.** Содержимое этой части окна зависит от режима, в котором вы сейчас работает. В режиме «Dialog» здесь отображаются элементы управления, принадлежащие активному «трехмерному» условному знаку (см. рис. А.8.). В режиме «Script» на этом месте отображается скрипт, принадлежащий активному «трехмерному» условному знаку, а также сообщения от компилятора (см. рис. А.9.).

**Содержимое «Конструктора» в режиме «Script» (см. рис. А.9.):**

1. **Список входных параметров.** Этот список не может быть отредактирован. Он состоит из обязательных параметров и других параметров, которые зависят от элементов управления принадлежащий активному знаку (см. рис. А.10);.

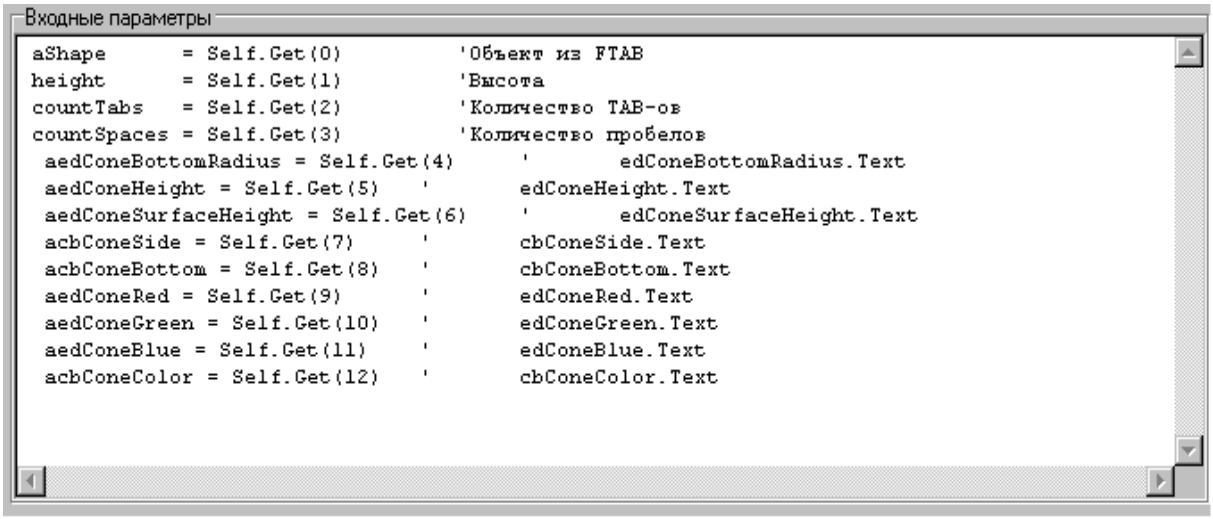



Рисунок А.10. Входные параметры знака «Cone»

2. **Тело скрипта.** Этот текст может быть отредактирован. Скрипт должен обязательно возвращать строку, соответствующую правильному описанию 3D модели на языке VRML (см. рис. А.11.);

Тело скрипта

```
aMaterial = {{"diffuseColor" ,aR+"  "+aG+"  "+aB}}
aAppearance = {{"material",aMaterial}}
if(aedConeBottomRadius.IsNumber=false)
  then aedConeBottomRadius = "2"
end
if(aedConeHeight.IsNumber=false)
  then aedConeHeight = "5"
end
if (aedConeSurfaceHeight.IsNumber)
  then aHeight = aedConeSurfaceHeight
  else aHeight= "0"
end

aHeight = aHeight.AsNumber+(aedConeHeight.AsNumber/2)
aPoint = {aShape.GetX,aShape.GetY}
tIndex = av.Run("World.LocalPoint",aPoint)
aHeight = aHeight+av.Run("World.GetPointHeight",{aPoint,tIndex})
aHeight = aHeight.AsString
aCoord = aShape.GetX.AsString+" "+aHeight+" "+(aShape.GetY.Negate).AsString
aCone = {{"height",aedConeHeight},{"bottomRadius",aedConeBottomRadius},{"side",acbConeSide}
aObject = {{"appearance",aAppearance},{"geometry","Cone",aCone}}
aTransform ={{"children",{{"Shape",aObject}}},{"translation",aCoord}}
Return av.Run("World.Node_Transforml",{TAB,"",aTransform})
```

Рисунок А.11. Тело скрипта знака «Cone»

3. **Сообщения компилятора.** В это поле выводится описание первой из встреченных ошибок, либо если ошибок нет, то выводится сообщение, что компиляция прошла успешно.

**Совет:** *для того чтобы компилировать скрипты необходимо, чтобы был открыт проект ArcView, желательно чтобы это был новый проект.*

К сожалению, компилятор Avenue не проверяет у объектов наличия вызываемых методов, если метода нет, то такая ошибка обнаружится только во время выполнения. Поэтому сообщение «Компиляция прошла успешно» не означает, что ошибок нет.

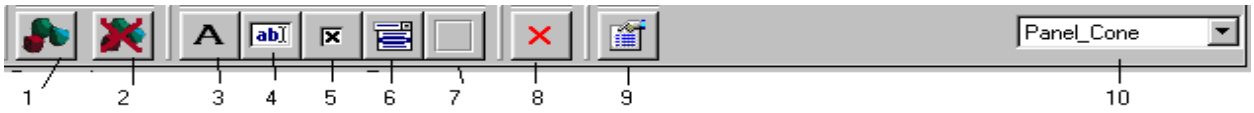



Рисунок А.12. Панель инструментов в режиме «Dialog».

**Содержимое панели инструментов в режиме «Dialog» (см. рис. А.12.):**

1. Кнопка, новый знак. Создает новый условный знак.
2. Кнопка, удалить знак. Удаляет существующий условный знак.
3. Кнопка, добавить новый элемент управления TLabel.
4. Кнопка, добавить новый элемент управления TEdit.
5. Кнопка, добавить новый элемент управления TCheckBox.

6. Кнопка, добавить новый элемент управления TComboBox.
7. Кнопка, добавить новый элемент управления TPanel.
8. Кнопка, удалить выбранный элемент управления.
9. Кнопка, свойства активного элемента управления, имя активного элемента управления показано в выпадающем списке “10”. В зависимости от типа элемента управления открывается форма, позволяющая изменять свойства этого элемента управления.
10. Выпадающий список, показывает все элементы управления активного условного знака.


Рисунок А.13. Панель инструментов в режиме «Script»

**Содержимое панели инструментов в режиме «Dialog» (см. рис. А.13.):**

1. Кнопка, компилировать скрипт. Позволяет компилировать скрипт .

### Создание и модификация элементов управления

Манипулировать с элементами управления можно только в режиме «Dialog». В этом режиме в левой части окна вы можете видеть элементы управления активного условного знака (см. рис.А.14). У каждого «трехмерного» условного знака существует элемент управления TPanel по умолчанию. Эта панель не может быть удалена. Имя этой панели представляет собой «Panel_»+<имя_знака>. Все остальные элементы управления расположены на этой панели.

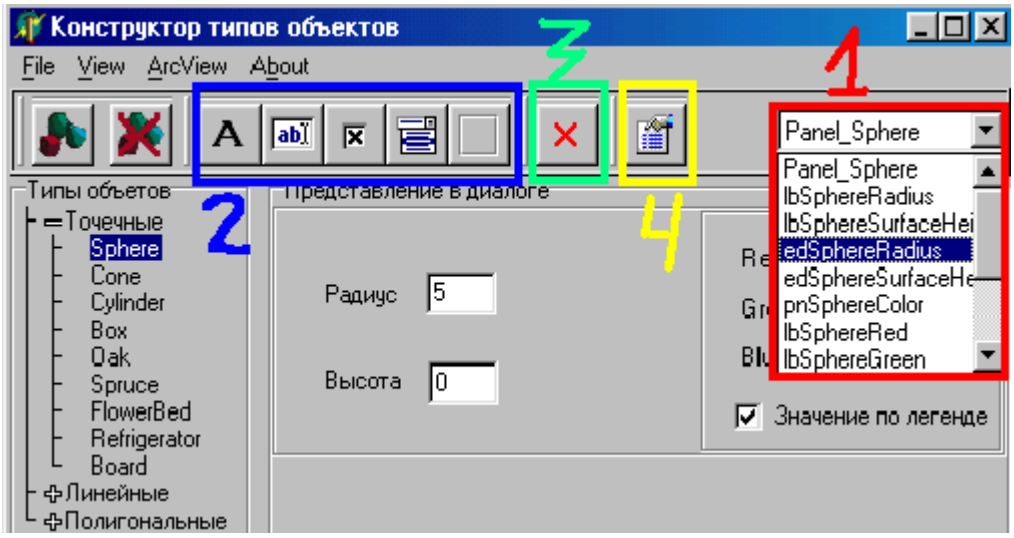



Рисунок А.14. Манипуляция с элементами управления.

В каждый момент времени лишь один элемент управления является активным. Имя активного в данный момент элемента управления вы можете видеть в выпадающем списке «1». Чтобы изменить активный элемент управления нужно в выпадающем списке

«1», выбрать имя нужного элемента управления, также можно просто щелкнуть курсором мыши на нужном элементе управления. Когда вы наводите курсор мыши на элемент управления, он изменится на «руку».

Переместить элемент управления, можно щелкнув на нем мышью, далее удерживая кнопку, перетащите элемент управления на нужное место.

Для создания нового элемента управления нажмите одну из кнопок «2». При этом откроется диалог, в котором нужно будет указать имя нового элемента управления, а также имя родительского элемента управления (см. рис. А.15.).

**Совет:** *имена элементам управления лучше давать осмысленные.*

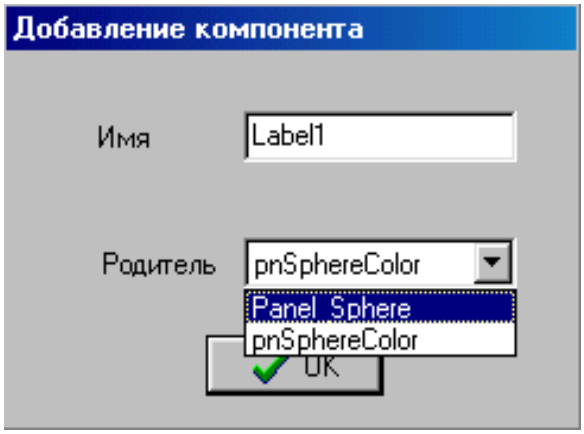



Рисунок А.15. Создание элемента управления.

Нажав кнопку «3» вы удалите активный элемент управления.

Нажав кнопку «4» откроется диалог, в котором вы сможете настроить свойства активного элемента управления, свойства будут зависеть от типа элемента управления. Например, на рисунке А.16. вы можете видеть диалог настройки свойств для элементов управления типа TPanel.

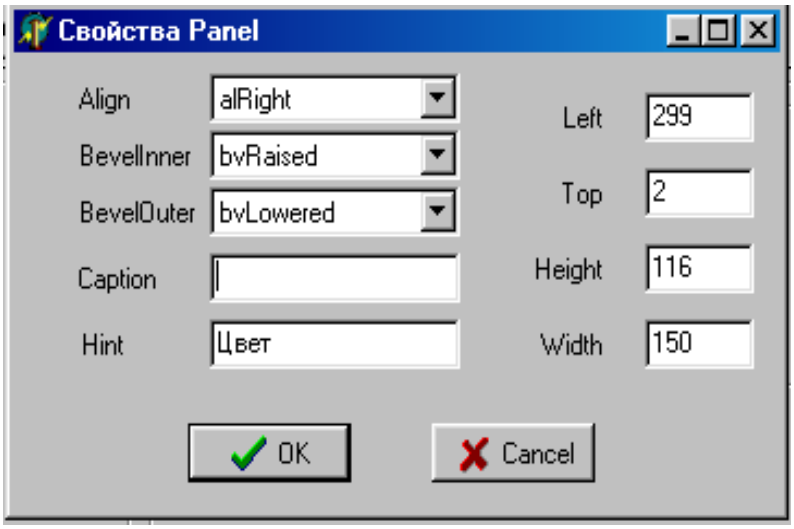



Рисунок А.16. Свойства TPanel.

Здесь необходимо отметить, что свойства некоторых элементов управления становятся входными параметрами скриптов. Например, для элементов управления типа

TEdit это свойство Text, для элементов управления типа TCheckBox это свойство Checked. Однако элементы управления типа TLabel или TPanel не передают в скрипт параметров.

### Создание, модификация скрипта

Для того чтобы создать новый «трехмерный» условный знак вам обязательно необходимо написать скрипт.

Все скрипты соответствующие условным знакам хранятся в папке **..\Scripts**. Также в папке **..\Scripts\System** хранятся «системные» скрипты.

Каждый скрипт представляет собой текстовый файл состоящий:

1. Число – определяет сколько далее следует строк параметров передающихся в данный скрипт. Зависят от элементов управления условного знака. Для «системных» скриптов это 0;
2. Строки параметров;
3. Тело скрипта.

При инсталляции в проект ArcView встраиваются все скрипты соответствующие всем имеющимся сейчас в библиотеке условным знакам, имена этих скриптов состоят из **«World.Object.»+<имя_знака>**. Также в проект встраиваются «системные» скрипты. Это скрипты выполняющие какие-либо операции. Имя этих скриптов складывается из **«World.»+<имя_файла>**.

Вы можете вызывать из вашего скрипта системные скрипты (например, вам нужно найти точку пересечения двух отрезков), чтобы облегчить вам написание кода. Посмотрите описание всех системных скриптов в «Руководстве программиста», чтобы узнать, какие функции уже реализованы. Также во время работы создаются глобальные переменные, которые хранят различную, важную информацию, которая может вам понадобиться.

**Нерешенная проблема.** *Экспериментальным путем установлено, что встраивание в ГИС ArcView слишком больших скриптов приводит к тому, что ArcView не может открыть потом ни одну таблицу, проект* ***«умирает»****. Встраивание скриптов происходит на основе механизма DDE. В документации по DDE ограничений на размер передаваемых данных нет. Я пришел к выводу, что это ошибка ГИС ArView.*

Решением можно преложить не очень удачное, но оно работает. Разбивать большие скрипты на более мелкие. В данной работе не предусмотрено приложения для создания «системных» скриптов. Создать новый «системный» скрипт можно в любом текстовом редакторе. Затем перенесите созданный файл в директорию **..\Scripts\System** и можете его использовать.

Далее, в качестве примера, приведу решение такой задачи. Взять значение «количество этажей» для условного знака «building» из поля таблицы. Для этого нужно:

- провести классификацию по полю «этажность». Данная библиотека поддерживает четыре типа легенды: Single Symbol, Graduated Symbol, Graduated Color, Unique Value;
- поставить для каждого класса значение высоты в поле Label;

- в скрипте значение высоты можно взять из глобальной переменной _aClass. _aClass это список содержащий:
    1. Label - строка;
    2. Red - значение (0..255) компоненты цвета;
    3. Green - значение (0..255) компоненты цвета;
    4. Blue  - значение (0..255) компоненты цвета.

Строка будет примерно такой aHeight = _aClass.Get(0).

При возникновении вопросов присылайте их мне по адресу kryazhev@ngs.ru.

## Приложение В. Руководство программиста

В этом приложении будут описаны все модули созданные в ходе данной работы, а также указана их взаимосвязь.

Большинство диалогов используемых в данной работе были реализованы на Delphi6.0. Все модули очень тесно интегрированы. В этом приложении модули будут перечислены, а также будет описана общая схема их взаимодействия.

Вся работа в целом может быть названа: «Создание библиотеки трехмерных условных знаков». Общая схема работы представлена на рисунке В.1.

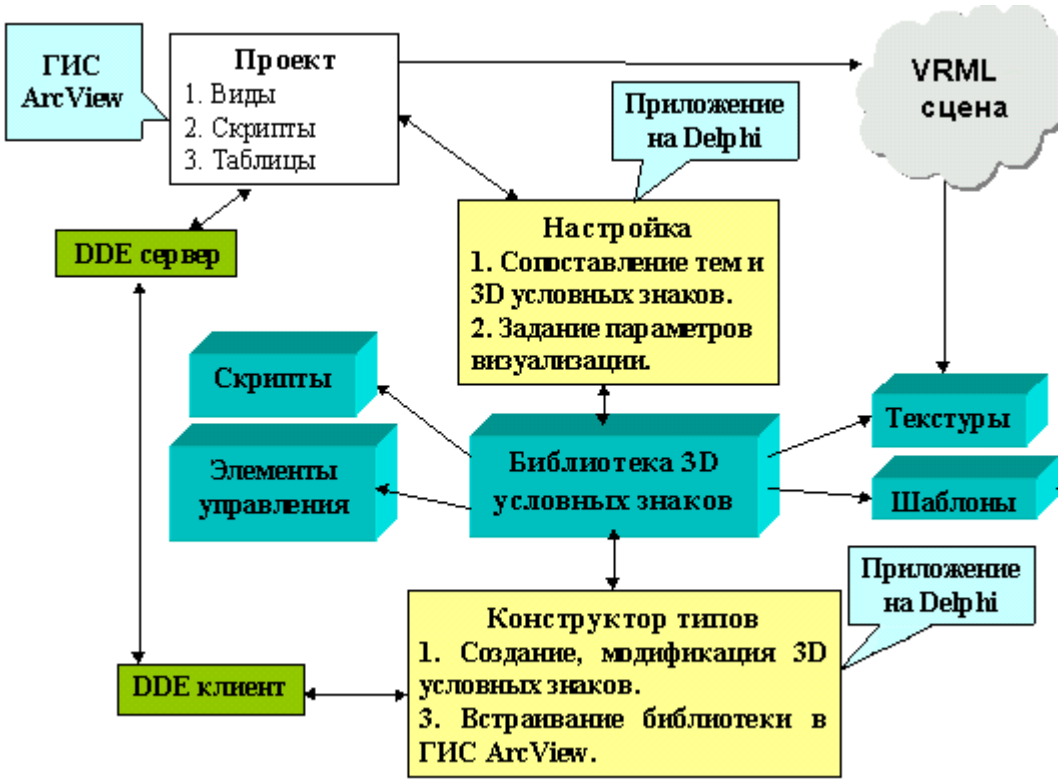



Рисунок В.1. Схема работы с библиотекой «трехмерных» условных знаков.

Под термином «библиотека» подразумевается следующая совокупность каталогов:

- ..**Scripts\.** В этом каталоге хранятся скрипты реализованные на языке Avenue, которые генерируют строки соответствующие правильным конструкциям на VRML. Эти скрипты «Конструктор типов» встраивает в проект ArcView при инсталляции. В этом каталоге есть подкаталог **..\System**\, в котором хранятся «системные» скрипты, также встраиваемые во время инсталляции в проект, эти скрипты обеспечивают общее взаимодействие или реализуют полезные функции. Например, скрипты для работы с триангуляцией. Также посмотрите раздел «Структура файла хранящего скрипт»;
- ..**Types\.** Здесь хранятся файлы, описывающие элементы управления. Эти элементы управления будут отображаться в диалоге «Настройка визуализации». Каждый условный знак имеет элементы управления, благодаря

которым пользователь может настраивать параметры генерируемого условного знака;

- ..**Tex\.** Здесь хранятся текстуры в формате который поддерживает язык VRML. Некоторые условные знаки могут использовать текстуры;
- ..**Wrl\.** Здесь хранятся готовые файлы в формате VRML. Шаблоны. Некоторые условные знаки могут не создавать новое описание 3D объекта, а ссылаться на уже готовые VRML файлы. Здесь могут находиться файлы в формате VRML1.0, VRML2.0 либо сжатые файлы VRML.

Библиотека это совокупность «трехмерных» условных знаков. Каждый условный знак представляет собой:

- Файл **<Имя_знака>.txt** . Файл находится в каталоге **..\Scripts**\.и содержит скрипт на языке Avenue. В скрипте может быть указана ссылка на текстуры либо на готовые VRML файлы. Также посмотрите раздел «Структура файла хранящего скрипт».
- Файл **<Имя_знака>.obj** . Местоположение файла зависит от того для какого типа картографического условного знака создан этот «трехмерный» условный знак (**..\Types\Point\, ..\Types\Line\, ..\Types\Polygon\)**. Также посмотрите раздел «Структура файла хранящего элементы управления».

В ходе работы было разработано два приложения:

- «Конструктор типов». Этот проект обеспечивает следующие возможности:
  - инсталляцию библиотеки в проект ArcView;
  - добавление в библиотеку новых «трехмерных» условных знаков;
  - модификацию существующих в библиотеке знаков.
- Диалог «Настройка». Этот проект запускается из проекта ArcView. Этот проект обеспечивает следующие возможности:
  - Настройка общих параметров генерируемого VRML файла;
  - Настройка параметров визуализации модели поверхности;
  - Сопоставление тем и «трехмерных» условных знаков, а также настройка параметров визуализации 3D условных знаков.

## *Преобразование координат*

### *При генерации:*

- **значение X (ArcView) переходит в X (VRML);**
- **значение Y (ArcView) переходит в -Z (VRML);**

#### Структура файла хранящего скрипт

Файл представляют собой текстовый файл. Все строки кроме первой – это строки кода на языке Avenue. Первая строка файла содержит количество параметров передаваемых в скрипт, это сделано, чтобы при использовании этого файла в «Конструкторе типов» отделить «список параметров» от «тела скрипта». Для системных скриптов первая строка “0”.

При инсталляции все строки кроме первой будут передаваться в ArcView.

### Структура файла хранящего элементы управления

Файл представляют собой текстовый файл. В этом файле хранится описание элементов управления и их свойств. В данной работе было выбрано пять элементов управления:

- TLabel . Следующие свойства можно настраивать:
  1. Align;
  2. AutoSize;
  3. Caption;
  4. Hint;
  5. Left;
  6. Top;
  7. Heigth;
  8. Width.
- TEdit . Следующие свойства можно настраивать:
  1. AutoSize;
  2. Hint;
  3. MaxLength;
  4. Text;
  5. Left;
  6. Top;
  7. Heigth;
  8. Width.
- TComboBox . Следующие свойства можно настраивать:
  1. ItemIndex;
  2. Items;
  3. Style;
  4. Hint;
  5. Left;
  6. Top;
  7. Heigth;
  8. Width.
- TCheckBox . Следующие свойства можно настраивать:
  1. Checked;
  2. Caption;
  3. Hint;
  4. Left;
  5. Top;
  6. Heigth;
  7. Width.
- TPanel . Следующие свойства можно настраивать:
  1. Align;
  2. BevelInner;
  3. BevelOuter;
  4. Hint;
  5. Left;
  6. Top;
  7. Heigth;
  8. Width.

Структура файла следующая:

1. Размерность знака. **(“Точечные”, “Линейные”, “Полигональные”)**;
2. **Название** условного знака;
3. Список элементов управления:
    3.1. **ClassName** элемента управления;
    3.2. **Name** элемента управления;
    3.3. **Parent** элемента управления;
    3.4. **Список свойств элемента управления.** Зависит от типа. Списки свойств приведены выше.

### Структура файла хранящего настройки DDE сервера

Этот текстовый файл находится в каталоге ..**\Ini** и называется **settings.cfg**. Он содержит три следующие строки:

1. Путь до файла arcview.exe;
2. Свойство DDE сервера **Service**;
3. Свойство DDE сервера **Topic**.

### Структура файла хранящего путь до библиотеки

Этот файл называется **world.cfg** и хранит в себе полный путь до файла prDialog.exe (диалог «Настройка»). При инсталляции в проект ArcView этот файл копируется в каталог проекта.

### Структура файла хранящего параметры визуализации

Для обмена данными между видом (View) ArcView и диалогом «Настройка», а также для хранения параметров визуализации используется специальный текстовый файл. Имя у такого файла представляет собой **<Имя_вида>.cfg**. В этом файле хранятся общие настройки а. Файл имеет следующую структуру:

1. Количество общих параметров. В данной работе 14;
2. Путь до браузера;
3. Путь до выходного файла;
4. Путь до файла триангуляции;
5. Масштабный коэффициент по X;
6. Масштабный коэффициент по Y;
7. Viewpoints. Значение (TRUE / FALSE);
8. Background. Значение (TRUE / FALSE);
9. Боковая сторона триангуляции. Значение (TRUE / FALSE);

10 – 15. Зарезервированные строки;

16. Количество тем;
17. Зависит от количества тем. Для каждой темы указываются следующее:
    - Название темы;
    - Размерность темы. Значение (0,1,2);
    - Активна ли тема. Значение (TRUE / FALSE);
    - Название «трехмерного» условного знака;
    - Размерность «трехмерного» условного знака;
    - Количество параметров;
    - Список параметров содержащий:
        - Название параметра;
        - Значение параметра.

## Структура файла триангуляции

В данной работе в качестве модели данных представляющей поверхность была выбрана триангуляционная нерегулярная сеть (TIN). Данные хранятся в текстовом файле, который имеет следующую структуру:

| | |
|---|---|
| “TIN FILE” | - заголовок файла; |
| “Point Count” | - строка; |
| <число> | - количество точек; |
| “Triangle Count” | - строка; |
| <число> | - количество треугольников; |
| “Points” | - строка; |
| <три числа> | - точка x y z; |
| …….. | |
| “Triangles” | - строка; |
| <шесть чисел> | - три индекса точек, три индекса треугольников. |
| …….. | |

## Структура заголовочного файла триангуляции

Файл триангуляции может иметь заголовочный файл. В заголовочном файле хранятся настройки визуализации поверхности. Настройка происходит следующим образом:

4. Находится диапазон высот, ищется min и max высоты в триангуляции.
5. Этот диапазон разбивается на несколько уровней.
6. Для каждого уровня можно определить цвет, который будут иметь вершины данной высоты.

Заголовочный файл, текстовый файл имеющий следующую структуру:

| | |
|---|---|
| “Header file from TIN” | - заголовок файла; |
| <число> | - количество точек; |
| <число> | - количество треугольников; |
| <число> | - min высота в триангуляции; |
| <число> | - max высота в триангуляции; |
| <число> | - количество уровней; |
| <число> | - min высота уровня; |
| <число> | - max высота уровня; |
| <число> | - цвет высоты в данного уровня; |
| …… | - три числа, зависит от количества уровней. |

## Модули созданные с помощью Delphi 6.0

Юнит **UnitDialog**, В этом юните реализован диалог «Настройка». В качестве параметров этому диалогу передаются две строки:

1. Полный путь до файла <Название_Вида>.cfg. Из этого текстового файла считываются все параметры генерируемого VRML мира. Структуру этого файла смотрите в разделе «Структура файла хранящего параметры визуализации»;
2. Полный путь до каталога с файлом “prCreator.exe”. Здесь должен быть подкаталог ..\Types\ в котом находятся файлы с расширением “obj”, эти файлы описывают все элементы управления принадлежащие «трехмерному» условному знаку.

**Процедуры и функции юнита UnitDialog:**

- TTheme.Create – конструктор объекта Ttheme. Создает необходимые элементы управления для каждой темы;
- TTheme.Destroy- деструктор;
- TTheme.Clear - удаляет все связанные с темой элементы управления;
- TTheme.Init - инициализирует элементы управления значениями;
- TTheme.SetControlValue – инициализирует заданный элемент управления, заданным значением;
- TTheme.CheckParam – проверяет соответствие между заданными параметрами и элементами управления;
- TfrmDialog.GetThemeNumber – получает номер темы в списке тем;
- TfrmDialog.LoadPointObjects – загружает в сплывающий список «трехмерные» условные знаки для точечных объектов;
- TfrmDialog.LoadLineObjects – загружает в сплывающий список «трехмерные» условные знаки для линейных объектов;
- TfrmDialog.LoadPolygonObjects – загружает в сплывающий список «трехмерные» условные знаки для полигональных объектов;
- TfrmDialog.ThemesClear – очищает список тем;
- TfrmDialog.LoadConfig – считывает файл конфигурации, заполняя соответствующие элементы управления значениями и создавая объекты TTheme;
- TfrmDialog.SaveConfig – сохраняет файл конфигурации;
- TfrmDialog.FormCreate – конструктор формы , происходит необходимая инициализация переменных;
- TfrmDialog.acChangeThemeExecute – смена выделенной темы;
- TfrmDialog.acChangeTypeExecute – смена «трехмерного» условного знака на другой, при этом элементы управления старого знака удаляются, создаются элементы управления нового условного знака и происходит их инициализация значениями по умолчанию;
- TfrmDialog.btbOkClick – сохраняются внесенные изменения и форма закрывается;
- TfrmDialog.btbCancelClick - форма закрывается не сохраняя изменений;
- TfrmDialog.btbApplyClick – сохраняются внесенные изменения;
- TfrmDialog.sbBrowserPathClick – открывается стандартный диалог (Open file) для указания пути до браузера;
- TfrmDialog.sbOutputPathClick – открывается стандартный диалог (Open file) для указания пути до выходного файла;
- TfrmDialog.leBrowserPathEnter – изменяет фокус ввода;
- TfrmDialog.edOutputPathEnter– изменяет фокус ввода;
- TfrmDialog.edTrianglesEnter– изменяет фокус ввода;
- TfrmDialog.btTrianglesClick – открывается диалог «Настройка триангуляции» для указания пути до файла триангуляции;

- TfrmDialog.cbActiveThemesClick – отображает все либо только активные темы.

Юнит **UnitTin**. В этом юните реализован диалог настройки работы с файлом триангуляции. Этот юнит используется в юните **UnitDialog**.

**Процедуры и функции юнита UnitTin:**

- RealToStr - преобразует значение типа double в строку;
- StrToReal – преобразует строку в значение типа double;
- TLevel.Create – создает и инициализирует объект типа TLevel;
- TLevel.Destroy – деструктор объекта типа TLevel;
- TfrmTIN_Settings.FormCreate – инициализирует переменные;
- TfrmTIN_Settings.FormClose – сохраняет параметры визуализации;
- TfrmTIN_Settings.TrianglesClear – сбрасывает все значения настроек;
- TfrmTIN_Settings.AddPoint – добавляет точку считанную из файла и определяет ее высоту;
- TfrmTIN_Settings.GetRange – вычисляет диапазон (от min до max) высот;
- TfrmTIN_Settings.acColorClickExecute – запускает стандартный диалог(Color Dialog), для изменения цвета диапазона;
- TfrmTIN_Settings.Exit1Click – выход из настроек;
- TfrmTIN_Settings.LevelsClear - сбрасывает все значения настроек;
- TfrmTIN_Settings.AddLevel – добавляет диапазон;
- TfrmTIN_Settings.RemoveLevel – удаляет диапазон;
- TfrmTIN_Settings.LoadTIN – считывает файл триангуляции;
- TfrmTIN_Settings.LoadHeader - считывает заголовочный файл триангуляции;
- TfrmTIN_Settings.SaveHeader - сохраняет заголовочный файл триангуляции;
- TfrmTIN_Settings.ChangeLevel – пересчитывает диапазоны уровней;
- TfrmTIN_Settings.edFromKeyPress – проверяет на допустимость нажатую клавишу;
- TfrmTIN_Settings.edFromChange - проверяет на допустимость введенное значение для начальных значений диапазонов;
- TfrmTIN_Settings.edToChange - проверяет на допустимость введенное значение для конечных значений диапазонов;
- TfrmTIN_Settings.udCountClassesChangingEx – изменяет значение уровней и пересчитывает диапазоны;
- TfrmTIN_Settings.tbLoadClick – открывает файл триангуляции;
- TfrmTIN_Settings.tbSaveClick – сохраняет настройки визуализации;
- TfrmTIN_Settings.tbClearClick – сбрасывает все значения настроек;
- TfrmTIN_Settings.Load1Click – открывает файл триангуляции;
- TfrmTIN_Settings.Save1Click – сохраняет настройки визуализации;
- TfrmTIN_Settings.Clear1Click – сбрасывает все значения настроек;
- TfrmTIN_Settings.edCountClassesEnter – изменяет фокус ввода.

Юнит **UnitLoadSave**. В этом юните реализованы процедуры для работы с файлом хранящим элементы управления. Этот юнит используется в юнитах: **UnitDialog, UnitCreator**.

**Процедуры и функции юнита UnitLoadSave:**

- LoadControls - считывает из файла параметры элементов управления, создает элементы управления и возвращает их в списке;
- SaveControls – сохраняет в файл список элементов управления.

Юнит **UnitCreator**, В этом юните реализован «Конструктор типов».

**Процедуры и функции юнита UnitCreator:**

- TfrmCreator.InstallScript - инсталлирует в проект ArcView скрипт;
- TfrmCreator.CompileScript - передает в проект ArcView скрипт и компилирует его;
- TfrmCreator.LoadScript - считывает скрипт из файла хранящего скрипт Avenue;
- TfrmCreator.SaveScript - сохраняет скрипт в файл;
- TfrmCreator.lbInvisibleMouseDown - перед началом перетаскивания элемента управления сохраняет начальную точку;
- TfrmCreator.lbInvisibleMouseMove - перетаскивание элемента управления в новую точку;
- TfrmCreator.lbInvisibleMouseUp - заканчивается процесс перетаскивания элемента управления;
- TfrmCreator.lbInvisibleClick – при щелчке на элементе управления этот элемент управления становится активным;
- TfrmCreator.memParamEnter – не позволяет редактировать список входных параметров скрипта переключая фокус ввода;
- TfrmCreator.reErrorEnter – не позволяет редактировать сообщения компилятора переключая фокус ввода;
- TfrmCreator.SetLabelBehavior – устанавливает обработчики событий для вновь созданного элемента управления класса TLabel;
- TfrmCreator.SetEditBehavior – устанавливает обработчики событий для вновь созданного элемента управления класса TEdit;
- TfrmCreator.SetCheckBoxBehavior – устанавливает обработчики событий для вновь созданного элемента управления класса TCheckBox;
- TfrmCreator.SetComboBoxBehavior – устанавливает обработчики событий для вновь созданного элемента управления класса TComboBox;
- TfrmCreator.SetPanelBehavior – устанавливает обработчики событий для вновь созданного элемента управления класса TPanel;
- TfrmCreator.SetControlsBehavior – устанавливает обработчики событий для списка созданных элементов управления;
- TfrmCreator.CreatePanel – создает элемент управления класса TPanel;
- TfrmCreator.AddObject – добавляет новый «трехмерный» условный знак;
- TfrmCreator.RemoveObject – удаляет «трехмерный» условный знак;
- TfrmCreator.AddComponent – добавляет новый элемент управления;
- TfrmCreator.RemoveComponent – удаляет активных элемент управления;
- TfrmCreator.ViewPanel – делает видимыми элементы управления активного условного знака, делая невидимыми все остальные элементы управления;
- TfrmCreator.ControlsDelete – удаляет для условного знака все элементы управления;
- TfrmCreator.ControlsClear – удаляет для всех условных знаков все элементы элементы управления;
- TfrmCreator.LoadObjects – считывает из файлов все имеющиеся в библиотеке условные знаки, эта процедура запускается при создании формы;

- TfrmCreator.SaveObject – сохраняет изменения внесенные в условный знак;
- TfrmCreator.SaveObjects – сохраняет в файлы все условные знаки, сохраняет внесенные в библиотеку изменения;
- TfrmCreator.FormCreate – инициализирует все глобальные переменные;
- TfrmCreator.AddLabel – добавляет активному условному знаку элемент управления TLabel;
- TfrmCreator.AddEdit – добавляет активному условному знаку элемент управления TEdit;
- TfrmCreator.AddCheckBox – добавляет активному условному знаку элемент управления TCheckBox;
- TfrmCreator.AddComboBox – добавляет активному условному знаку элемент управления TComboBox;
- TfrmCreator.AddPanel – добавляет активному условному знаку элемент управления TPanel;
- TfrmCreator.tbPropertyClick – выводит диалог для настройки активного элемента управления. Вид диалога зависит от класса элемента управления;
- TfrmCreator.tbAddClick – создает новый 3D условный знак;
- TfrmCreator.tbRemoveClick – удаляет 3D условный знак;
- TfrmCreator.tbDeleteControlClick – удаляет активный элемент управления;
- TfrmCreator.Dialog1Click – переключение в режим «Dialog»;
- TfrmCreator.Script1Click – переключение в режим «Script»;
- TfrmCreator.tbCompileClick – компилирует скрипт активного 3D условного знака;
- TfrmCreator.tbSave1Click – сохраняет изменения внесенные в 3D условный знак;
- TfrmCreator.tbSaveAll1Click – сохраняет все изменения внесенные в библиотеку;
- TfrmCreator.DdeServerSettingsClick – открывает диалог для настройки DDE сервера;
- TfrmCreator.tbSave1Click – сохраняет изменения внесенные в 3D условный знак;
- TfrmCreator.Install1Click – инсталлирует библиотеку в проект ArcView;
- TfrmCreator.FormDestroy – освобождает все используемые приложением ресурсы;
- TfrmCreator.otlObjectsClick – переключает активность на другой 3D условный знак.

Юнит **UnitIni**. В этом юните реализован диалог настройки параметров DDE сервера. Этот юнит используется в юните **UnitCreator**.

**Процедуры и функции юнита UnitIni:**

- TfrmSettings.FormCreate - считывает файл **settings.cfg**, хранящий настройки DDE сервера, инициализирует глобальные переменные;
- TfrmSettings.btPathClick - открывает стандартный диалог (Open file) для указания пути до файла **arcview.exe**;
- TfrmSettings.btDefaultClick - присваивает свойствам DDE сервера значения по умолчанию;
- TfrmSettings.FormClose - сохраняет в файл **settings.cfg**, параметры DDE сервера;
- TfrmSettings.btbOkClick - закрывает диалог.

Юнит **UnitCreateObject**. В этом юните реализован диалог создания нового условного знака. Здесь пользователь должен ввести имя нового условного знака, а также указать размерность знака. Этот юнит используется в юните **UnitCreator**.

**Процедуры и функции юнита UnitCreateObject:**

- TfrmCreateObject.FormCloseQuery - проверяет не существует ли уже знаков с таким именем, если имя не уникально, то выводится сообщение об ошибке.

Юнит **UnitAddComponent**. В этом юните реализован диалог создания нового элемента управления. Здесь пользователь должен ввести имя нового элемента управления, а также выбрать в выпадающем списке имя родительского компонента. Этот юнит используется в юните **UnitCreator**.

**Процедуры и функции юнита UnitAddComponent:**

- TfrmAddComponent.FormCreate - инициализирует переменные и содержимое элементов управления;
- TfrmAddComponent.FormCloseQuery - проверяет не существует ли уже элементов управления с таким именем, если имя не уникально, то выводится сообщение об ошибке;
- TfrmAddComponent.btbOkClick - закрывает диалог.

Также есть ряд юнитов реализующие диалоги настройки свойств для элементов управления. Названия процедур в этих юнитах одинаковы. Эти юниты отличаются только видом диалогов и реализацией процедур. Это следующие юниты: **UnitLabel, UnitEdit, UnitCheckBox, UnitComboBox, UnitPanel**. Все эти юниты используются в **UnitCreator**.

**Процедуры и функции юнитов:**

- actInputIntExecute - проверяет является ли нажатая клавиша цифрой;
- InputObject - при открытии диалога содержимое диалога инициализируется свойствами настраиваемого элемента управления;
- OutputObject - при закрытии диалога содержимое диалога переносится в свойства настраиваемого элемента управления;
- btbtOkClick - закрывает диалог настройки.

### Системные скрипты библиотеки реализованные на Avenue

**World.CheckRib** - находит индекс треугольника, к которому нужно переходить, при локализации точки.

Входные параметры:
- точка, которую локализуем;
- точка, центр треугольника в триангуляции;
- индекс треугольника в триангуляции, для которого производим проверку.

Выходные параметры:
- индекс треугольника в триангуляции.

**World.CheckSign** - проверяет, принадлежит ли точка прямой.

Входные параметры:
- точка, которую проверяем;
- две точки, концы отрезка.

Выходные параметры:
- если точка принадлежит прямой, то возвращается 0.

**World.CheckTriangle** - проверяет, принадлежит ли точка треугольнику.
Входные параметры:
- точка, которую проверяем;
- индекс треугольника.

Выходные параметры:
- булево значение.

**World.Compile** - компилирует скрипт.
Входные параметры:
- строка имя скрипта.

Выходные параметры:
- описание первой из встреченных ошибок.

**World.CompiledScripts** - проверяет, откомпилированы ли все скрипты с именем <World.*>, если нет, то выдается список всех скриптов содержащих синтаксические ошибки.

Выходные параметры:
- булево значение.

**World.DeleteFile** - удаляет файл.
Входные параметры:
- строка имя файла.

**World.ExecuteSettings** - обработчик кнопки «Настройка». Открывает диалог «Настройка», в котором производится настройка параметров визуализации.

**World.FindTheme** - находит в списке тем параметры визуализации нужной темы.
Входные параметры:
- строка название темы, которую будем искать;
- список тем, в котором будем искать.

Выходные параметры:
- описание первой из встреченных ошибок.

**World.GenerationConfig** - создает файл, в котором хранятся параметры генерируемого VRML файла.
Выходные параметры:
- список, содержащий общие настройки, а также настройки для каждой темы.

**World.GenerationObject** - этот скрипт создает список общих параметров и вызывает скрипт 3D условного знака, который должен сгенерировать VRML описание объекта, исходя из общих параметров.
Входные параметры:
- объект Shape из таблицы;
- список настроек для объектов данной темы.

Выходные параметры:
- строка, описание VRML сцены.

**World.GenerationOptions** - этот скрипт добавляет в выходной VRML файл источники освещения, точки наблюдения, узел NavigationInfo.

**World.GenerationSurface** - этот скрипт создает модель поверхности.
Входные параметры:
- имя файла триангуляции.

**World.GenerationSurfaceBottom** - этот скрипт добавляет «дно» к модели поверхности.

**World.GenerationSurfaceEx** - этот скрипт создает модель поверхности.

**World.GenerationSurfaceSide** - этот скрипт добавляет боковую сторону к модели поверхности.

**World.GenerationWorld** - этот скрипт обработчик события «GenerationWorld». *Это основной скрипт данной библиотеки. Во время его работы создается файл VRML.*

**World.GetArea** - вычисляет площадь треугольника.
Входные параметры:
- три точки.

Выходные параметры:
- площадь треугольника.

**World.GetClassification** - классифицирует объект по легенде.
Входные параметры:
- легенда темы;
- таблица темы;
- индекс объекта в таблице.

Выходные параметры:
- класс объекта.

**World.GetClassificationClass** - классифицирует объект по легенде.
Входные параметры:
- легенда темы;
- индекс класса объекта.

Выходные параметры:
- класс объекта.

**World.GetConstructorPath** - получает путь до «Конструктора» типов.

**World.GetIntersectLine** - вычисляет все точки пересечения отрезка с ребрами триангуляции.
Входные параметры:
- начальная точка;
- конечная точка;
- индекс треугольника (локализованная начальная точка).

Выходные параметры:
- список всех точек пересечения;
- список индексов треугольников(локализация каждой точки).

**World.GetIntersectPoint** - вычисляет точку пересечения двух прямых.
Входные параметры:
- четыре точки (концы отрезков).

Выходные параметры:
- точка пересечения.

**World.GetIntersectPolyline** - вычисляет все точки полилинии с ребрами триангуляции.
Входные параметры:
- полилиния.

Выходные параметры:
- список всех точек пересечения;

- список индексов треугольников (локализация каждой точки).

**World.GetPointHeight** - на основе барицентрических координат вычисляет z-координату.

Входные параметры:
- точка;
- индекс треугольника.

Выходные параметры:
- Z координата точки.

**World.GetPolygonHeight** - для списка точек(полигона) производит локализацию каждой точки и на основе барицентрических координат вычисляет z-координаты .

Входные параметры:
- полигон.

Выходные параметры:
- список, Z координаты точек.

**World.GetTriangleSelection** - производит вырезку треугольников из триангуляции, которые попали в ActiveExtent. В VRML файл заносятся только треугольники из этой вырезки.

**World.GetVertexColor** - вычисляет цвет вершины.

Входные параметры:
- индекс вершины.

Выходные параметры:
- цвет вершины.

**World.Install** - встраивает в проект ArcView все возможности библиотеки

**World.LocalPoint** - находит индекс тругольника в триангуляции, в который попадает точка.

Входные параметры
- точка, которую локализуем.

Выходные параметры
- индекс треугольника, в который попадает точка.

**World.NewTheme** - устанавливает настройки темы по умолчанию.

Входные параметры
- название темы.

Выходные параметры
- список параметров.

**World.Node_Appearance** - генерирует строку, соответствующую узлу Appearance в языке VRML.

**World.Node_Box** - генерирует строку, соответствующую узлу Box в языке VRML.

**World.Node_Cone** - генерирует строку, соответствующую узлу Cone в языке VRML.

**World.Node_Cylinder** - генерирует строку, соответствующую узлу Cylinder в языке VRML.

**World.Node_DirectionalLight** - генерирует строку, соответствующую узлу DirectionalLight в языке VRML.

**World.Node_DEF1** - генерирует строку, соответствующую DEF конструкции в языке VRML.

**World.Node_DEF2** - генерирует строку, соответствующую DEF конструкции в языке VRML.

**World.Node_DirectionalLight** - генерирует строку, соответствующую узлу **DirectionalLight** в языке VRML.

**World.Node_ImageTexture** - генерирует строку, соответствующую узлу **ImageTexture** в языке VRML.

**World.Node_IndexedFaceSet** - генерирует строку, соответствующую узлу IndexedFaceSet в языке VRML.

**World.Node_Inline** - генерирует строку, соответствующую узлу Inline в языке VRML.

**World.Node_Material** - генерирует строку, соответствующую узлу Material в языке VRML.

**World.Node_NavigationInfo** - генерирует строку, соответствующую узлу NavigationInfo в языке VRML.

**World.Node_PixelTexture** - генерирует строку, соответствующую узлу NavigationInfo в языке VRML.

**World.Node_Shape** - генерирует строку, соответствующую узлу Shape в языке VRML.

**World.Node_Sphere** - генерирует строку, соответствующую узлу Sphere в языке VRML.

**World.Node_TextureTransform** - генерирует строку, соответствующую узлу TextureTransform в языке VRML.

**World.Node_Transform1** - генерирует строку, соответствующую узлу Transform в языке VRML.

**World.Node_Transform2** - генерирует строку, соответствующую узлу Transform в языке VRML.

**World.Node_Transform3** - генерирует строку, соответствующую узлу Transform в языке VRML.

**World.Node_USE** - генерирует строку, соответствующую DEF конструкции в языке VRML.

**World.Node_Viewpoint** - генерирует строку, соответствующую узлу Viewpoint в языке VRML.

**World.OpenTin** - читает текстовый файл, содержащий триангуляцию.

Входные параметры

- имя файла.

Выходные параметры

- список точек;
- список треугольников;
- список диапазонов, определяет зависимость цвета вершины от ее высоты.

**World.OpenTinHeader** - читает заголовочный файл триангуляции.

Входные параметры
- имя файла.

Выходные параметры
- список диапазонов, определяет зависимость цвета вершины от ее высоты.

**World.ReadFromFile** - читает файл конфигурации ".cfg", содержащий общие настройки и настройки тем.

Входные параметры
- имя файла.

Выходные параметры
- общие настройки и настройки тем.

**World.RemoveScript** - удаляет скрипт из проекта.

Входные параметры
- имя скрипта.

**World.SaveToFile** - сохраняет файл конфигурации ".cfg", содержащий общие настройки и настройки тем.

Входные параметры
- имя файла;
- список общих настроек;
- список настроек тем.

**World.SaveToWorld** - сохраняет в VRML файл строку.

Входные параметры
- строка.

**World.Uninstall** - удаляет из проекта все ранее встроенные возможности, такие как: скрипты, кнопки, пункты меню.

## Глобальные переменные

При генерации 3D условного знака может возникнуть необходимость получить данные, которые не передаются в скрипт непосредственно (например, получить какую-нибудь атрибутивную информацию из таблицы). В ходе работы библиотеки инициализируются глобальные переменные, значение из них могут быть использованы для получения необходимой информации.

**_aProgramPath** – полный путь до файла prDialog.exe (диалог «Настройка»).

**_ aProgramDir** – полный путь до главного каталога библиотеки. Отсюда берутся текстуры (каталог **..\Tex**), а также шаблоны (каталог **..\Wrl**).

**_aView** – объект типа Doc. Активный вид.

**_aExtent** – объект типа Rect. Он строится как минимальный прямоугольник содержащий все объекты выборки (ActiveExtent), затем он расширяется на 50 метров.

**_aCenterX** – объект типа Number. Центр _aExtent (значение X).

**_aCenterZ** – объект типа Number. Центр _aExtent (значение Y).

**_aWidth** – объект типа Number. Ширина _aExtent(значение X).

**_aHeight** – объект типа Number. Высота _aExtent (значение Y).

**_aWorld** – это полный путь до выходного VRML файла. При необходимости можно записывать генерируемую строку непосредственно в него, при этом скрипт в операторе Return должен возвращать пустую строку.

**_aTable** – переменная хранит ссылку FTab.

**_aRecord** – это ссылка на запись в таблице.

**********

Пример (получение значения из поля с именем “SomeField”)

```
myField = _Table.FindField("SomeField")
aValue = _aTable.ReturnValue(myField,_aRecord)
```

**********

**_aClass** – глобальная переменная представляет собой список. Определяем к какому классу принадлежит данный объект (классифицируем по легенде). Список представляет собой:

- значение Label, для соответствующего класса.
- список {Red, Green, Blue}цветовых компонент, для соответствующего класса.

**_aPoints** – список всех точек триангуляции.

**_aSelPoints** – список точек триангуляции, которые попадают в _aExtent (вырезка из _aPoints).

**_aTriangles** – список всех треугольников триангуляции.

**_aSelTriangles** – список треугольников триангуляции, вершины которых попадают в _aExtent (вырезка из _aTriangles). Модель поверхности визуализируется на основании этих треугольников.

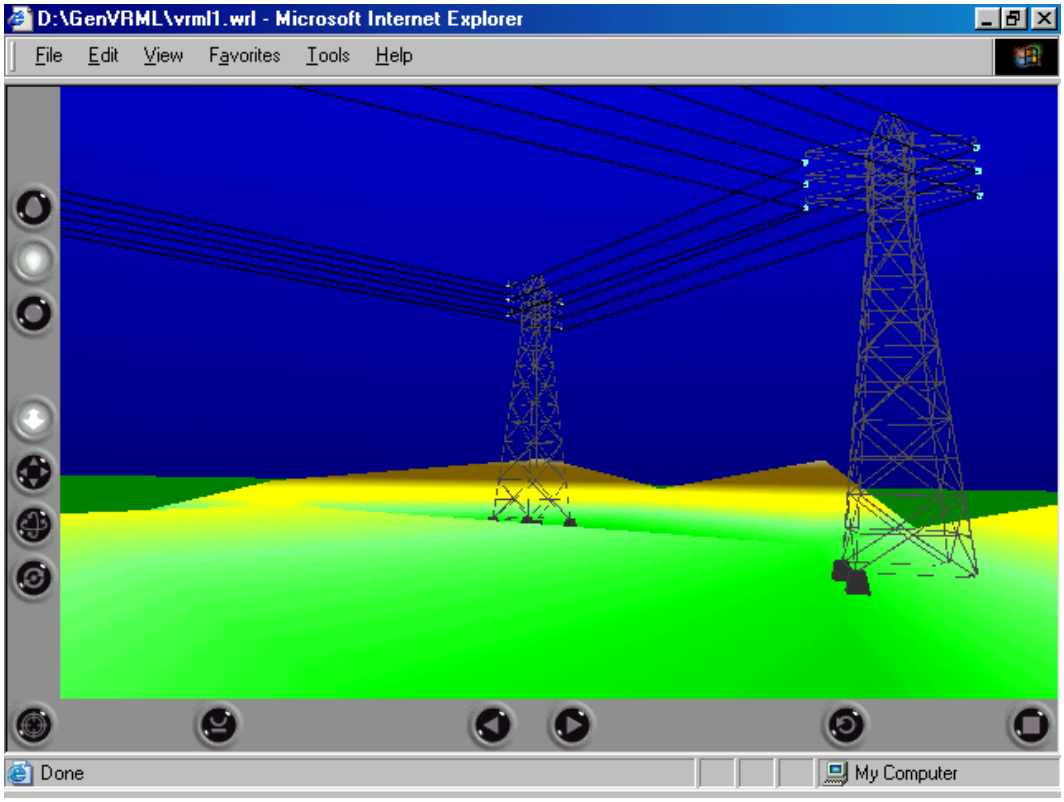
**_aMax** – максимальная высота вершины в файле триангуляции (если файл триангуляции не указан, то значение равно 0).

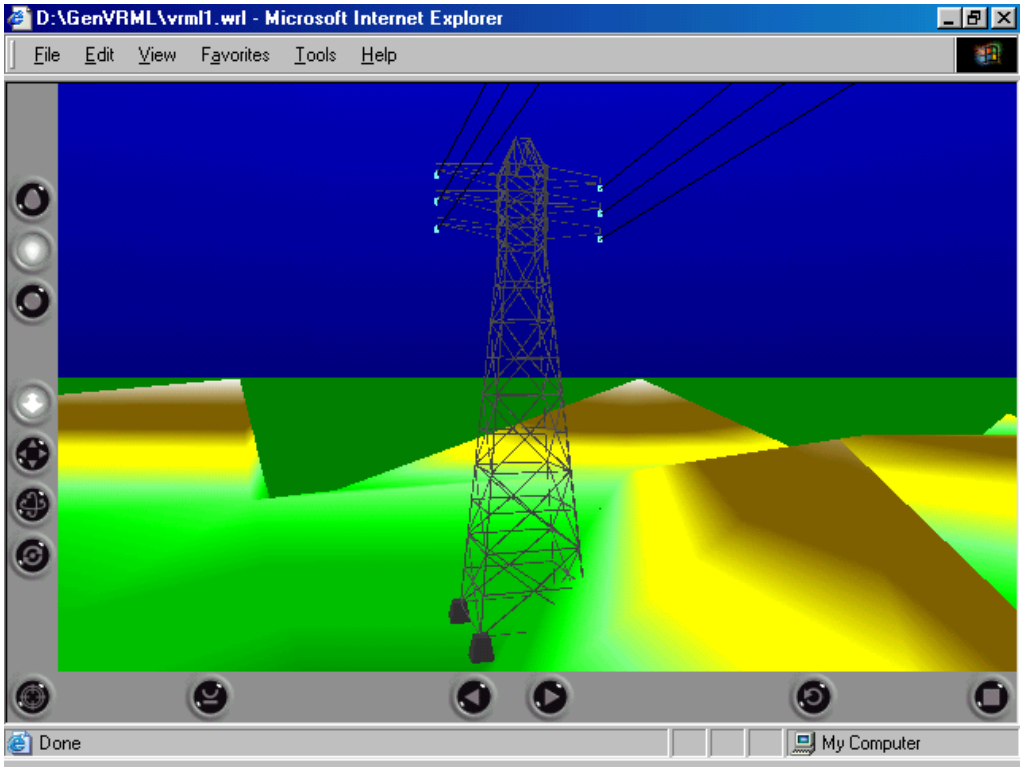
**_aMin** – минимальная высота вершины в файле триангуляции (если файл триангуляции не указан, то значение равно 0).

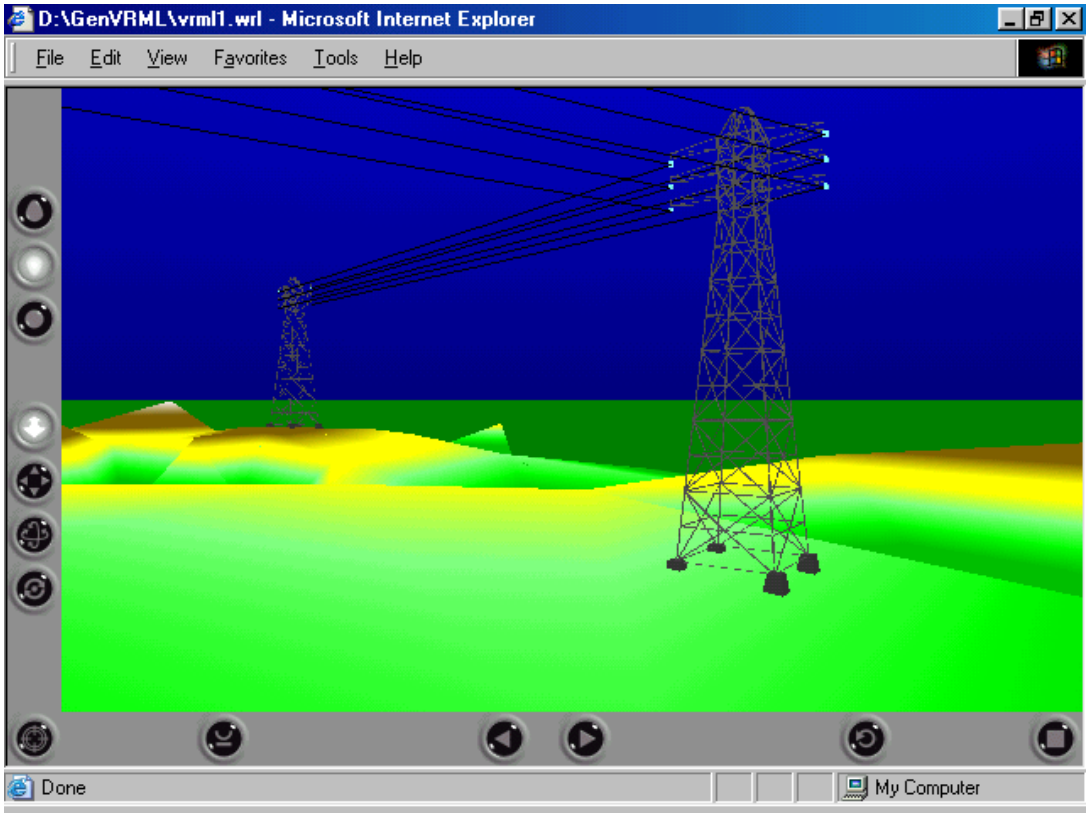
**_aLevels** – список, хранящий цветовую шкалу. Цвет вершины в ЦМР определяется на основе этого списка. Список представляет собой:

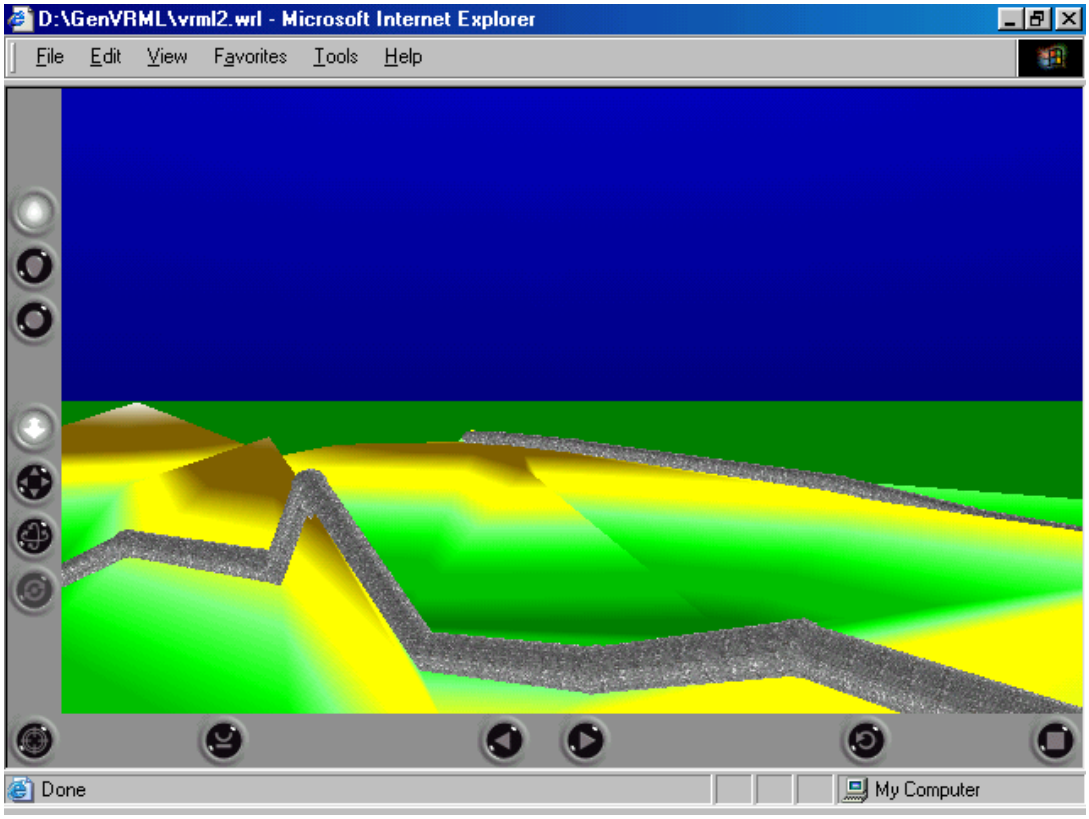
- значение высоты;
- red компонента цвета, для данной высоты;
- green компонента цвета, для данной высоты;
- blue компонента цвета, для данной высоты.

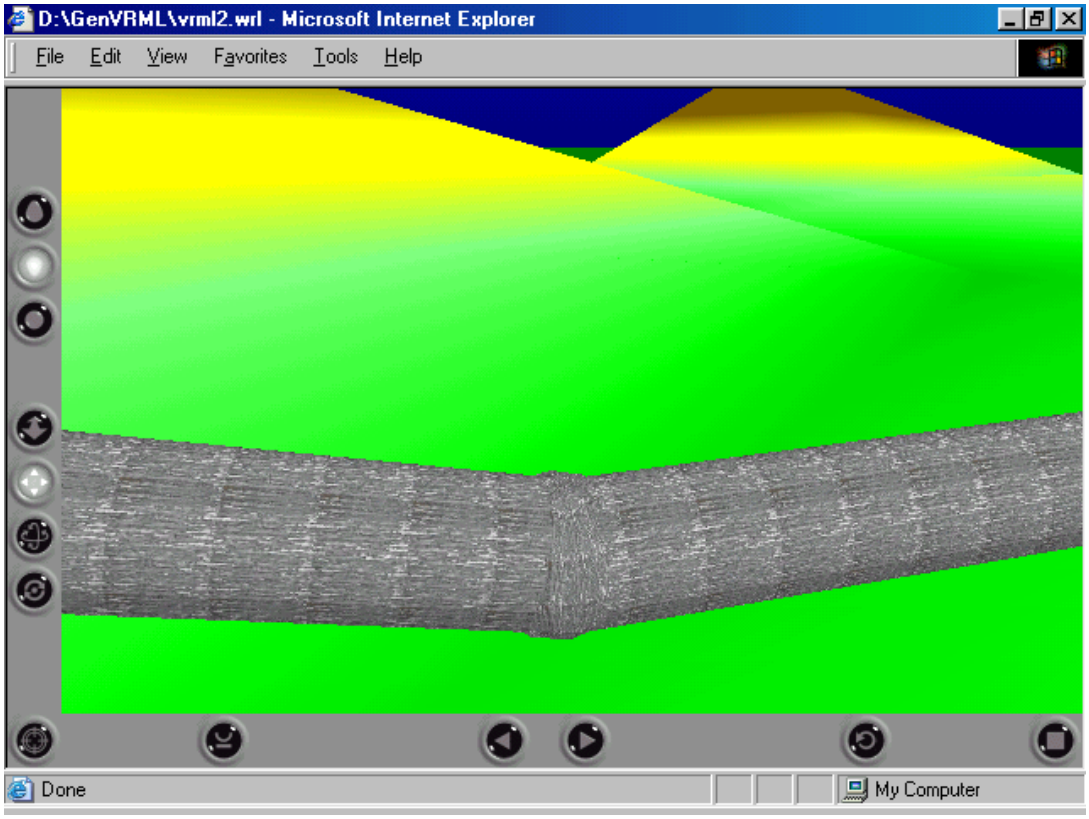
# Приложение С. Иллюстрации к дипломной работе

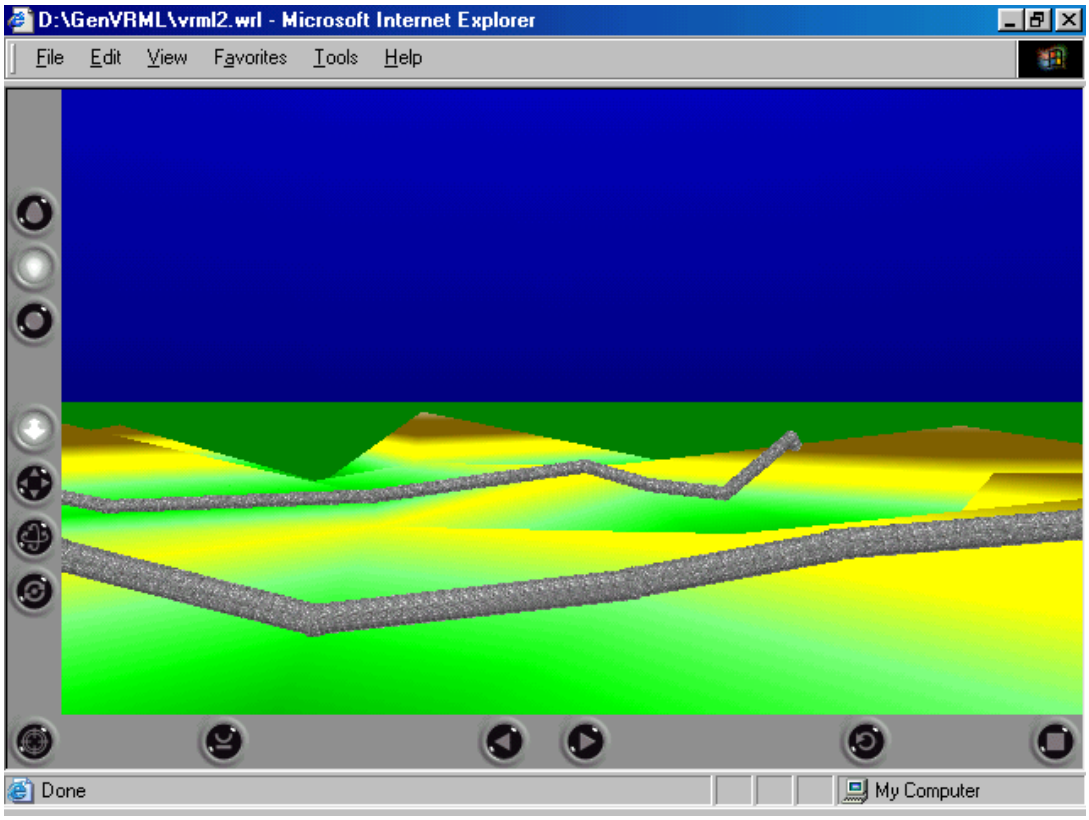


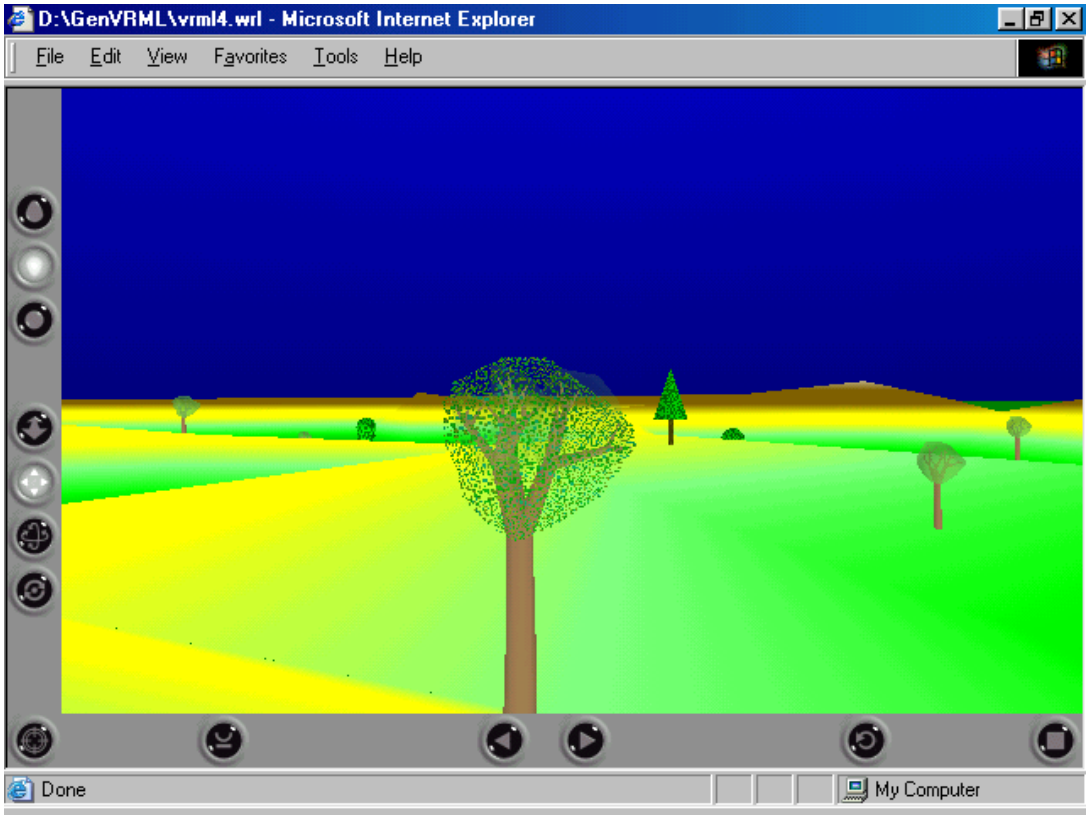



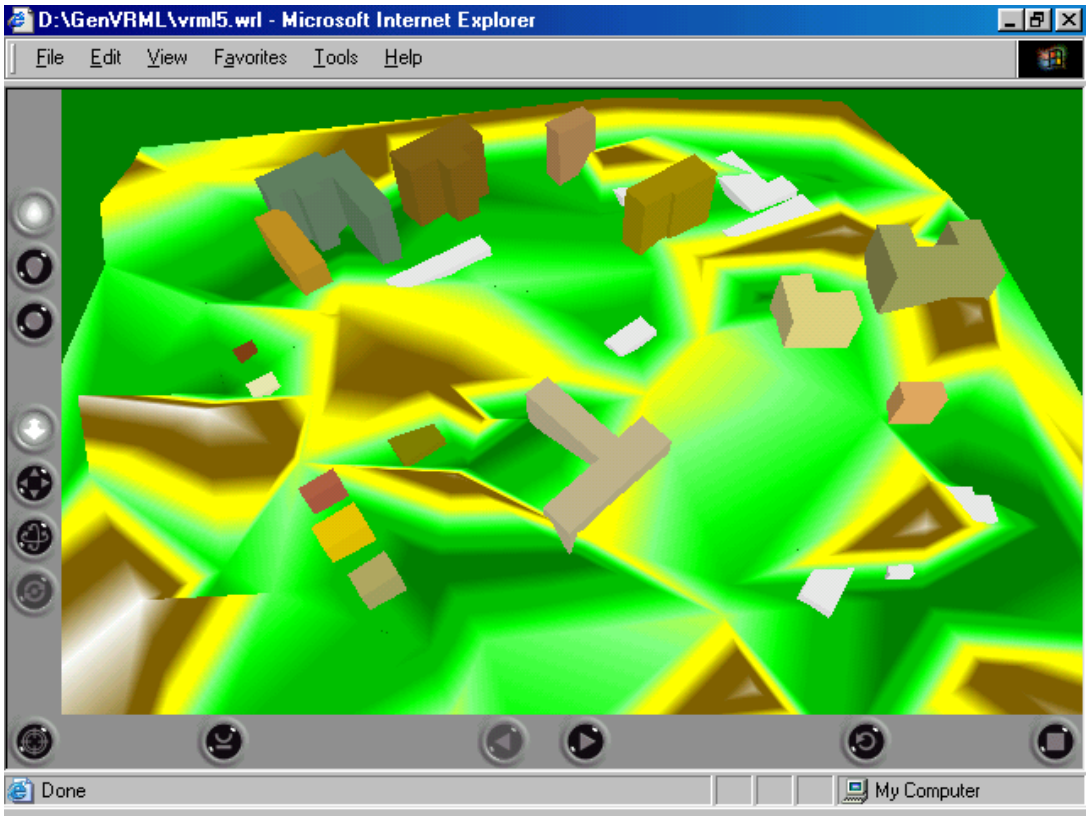

D:\GenVRML\vrml1.wrl - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Done
My Computer

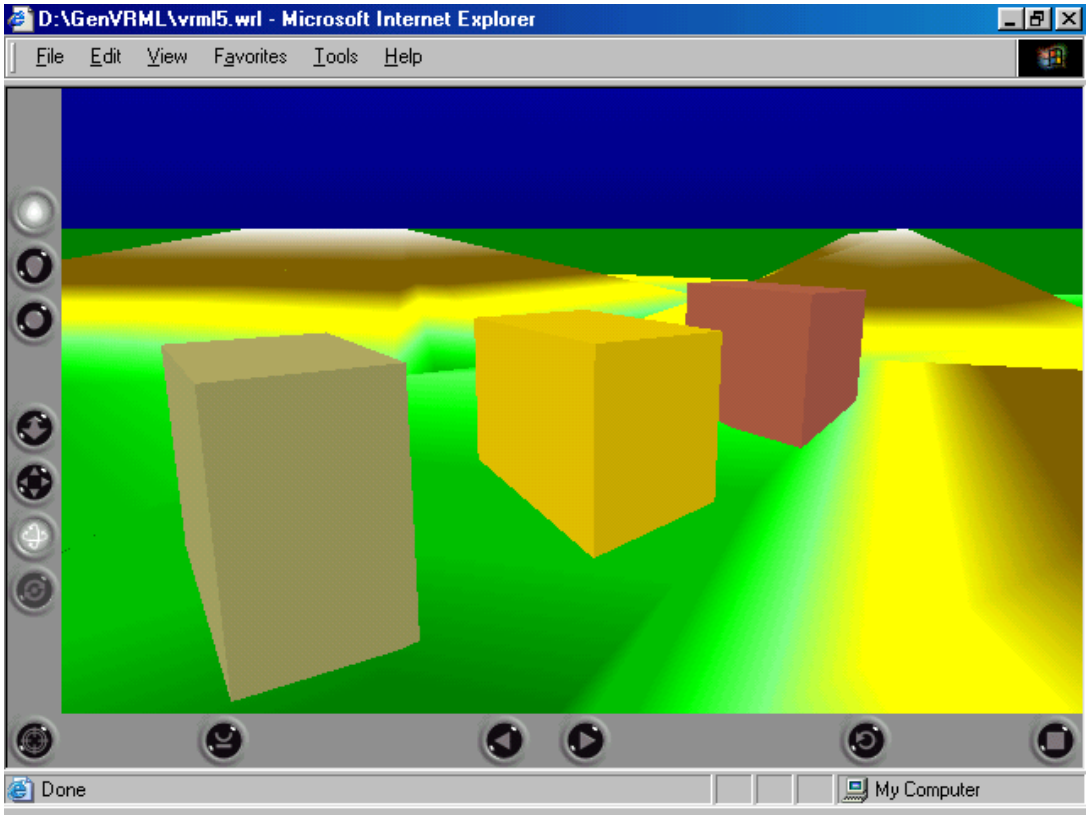

D:\GenVRML\vrml2.wrl - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Done
My Computer

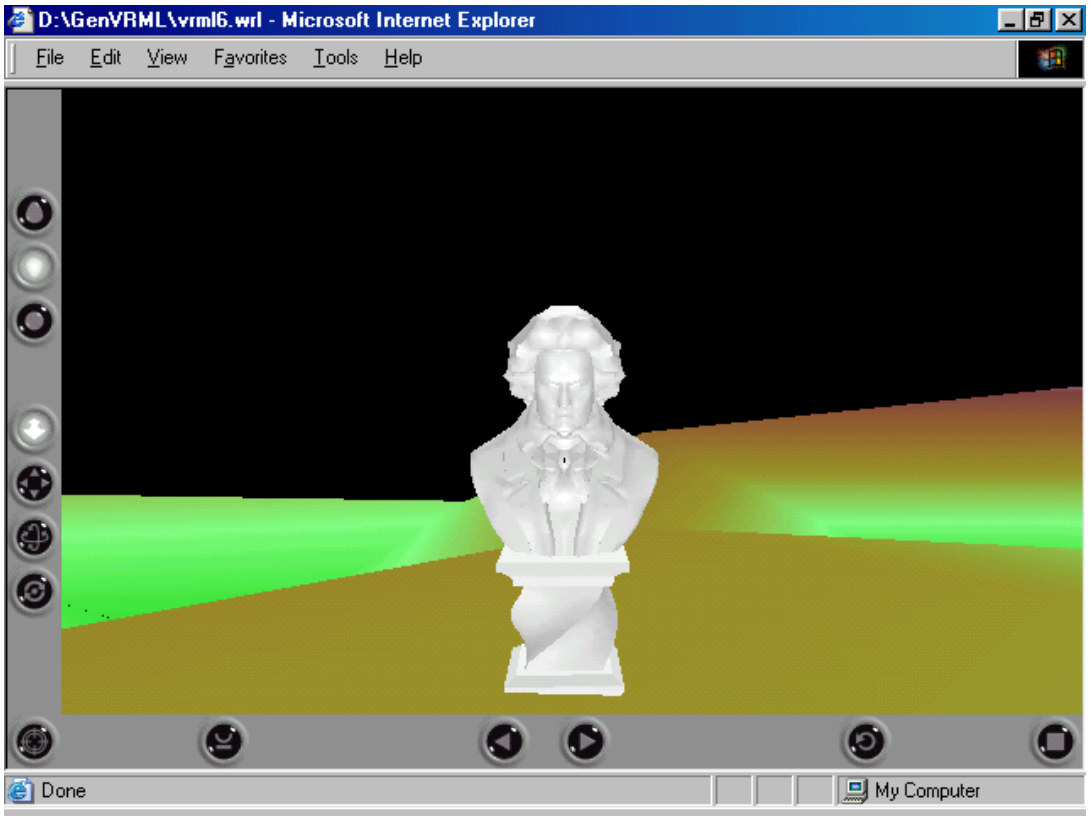

D:\GenVRML\vrml2.wrl - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Done
My Computer

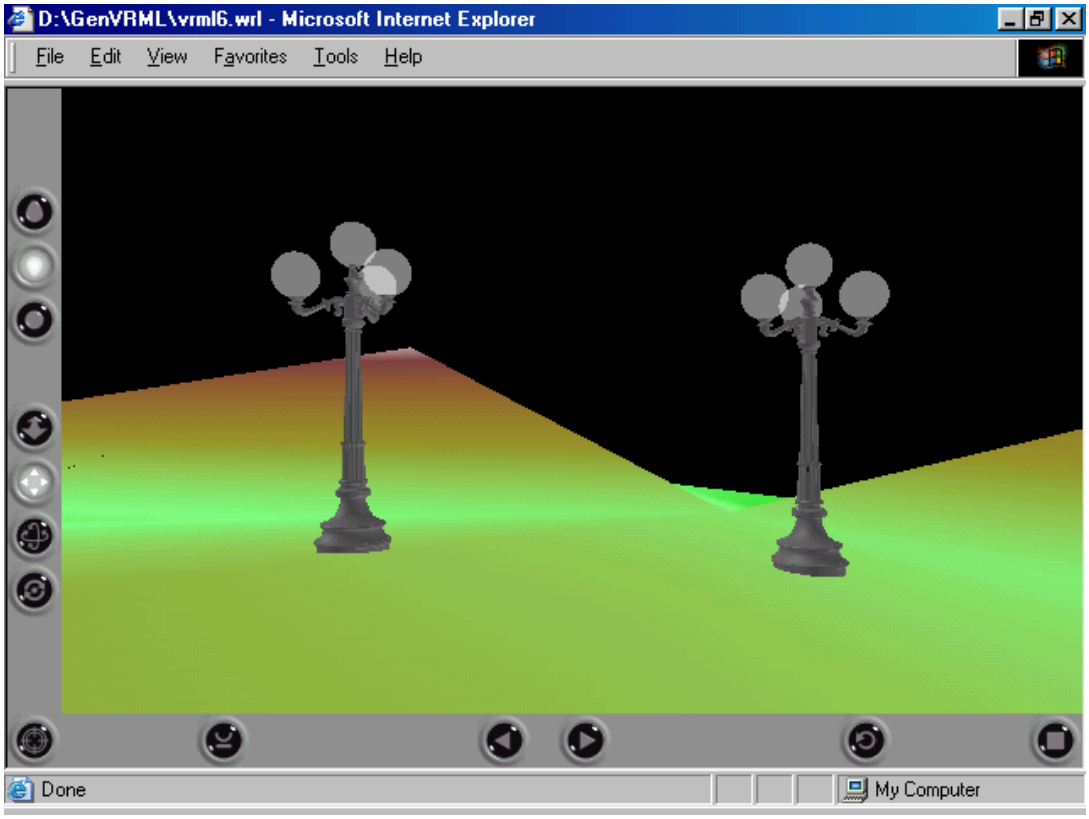

D:\GenVRML\vrml2.wrl - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Done
My Computer

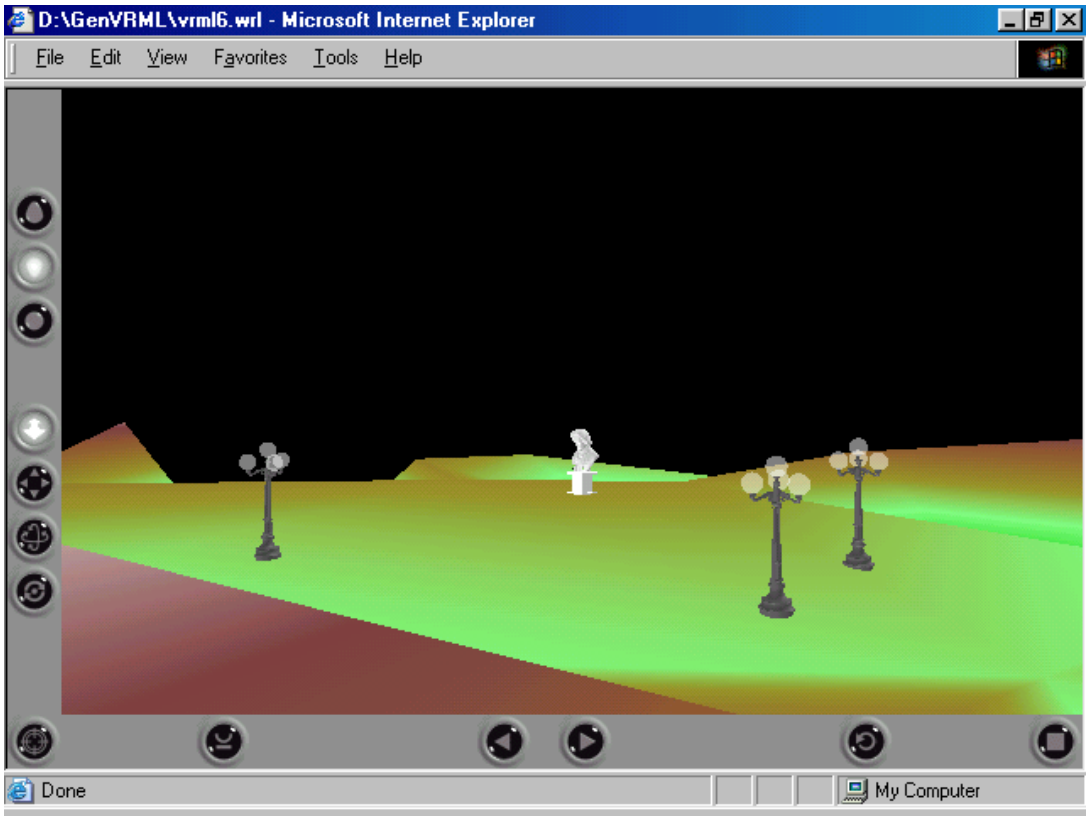

D:\GenVRML\vrml4.wrl - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Done
My Computer

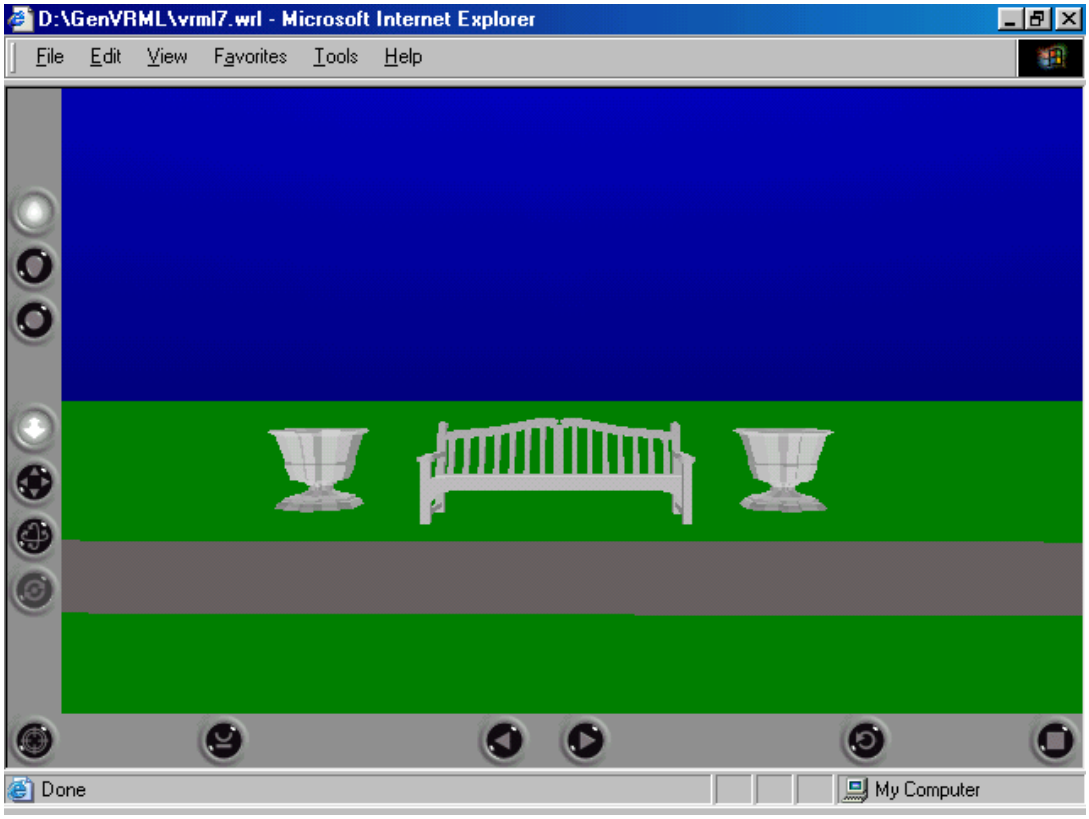

D:\GenVRML\vrml5.wrl - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Done
My Computer

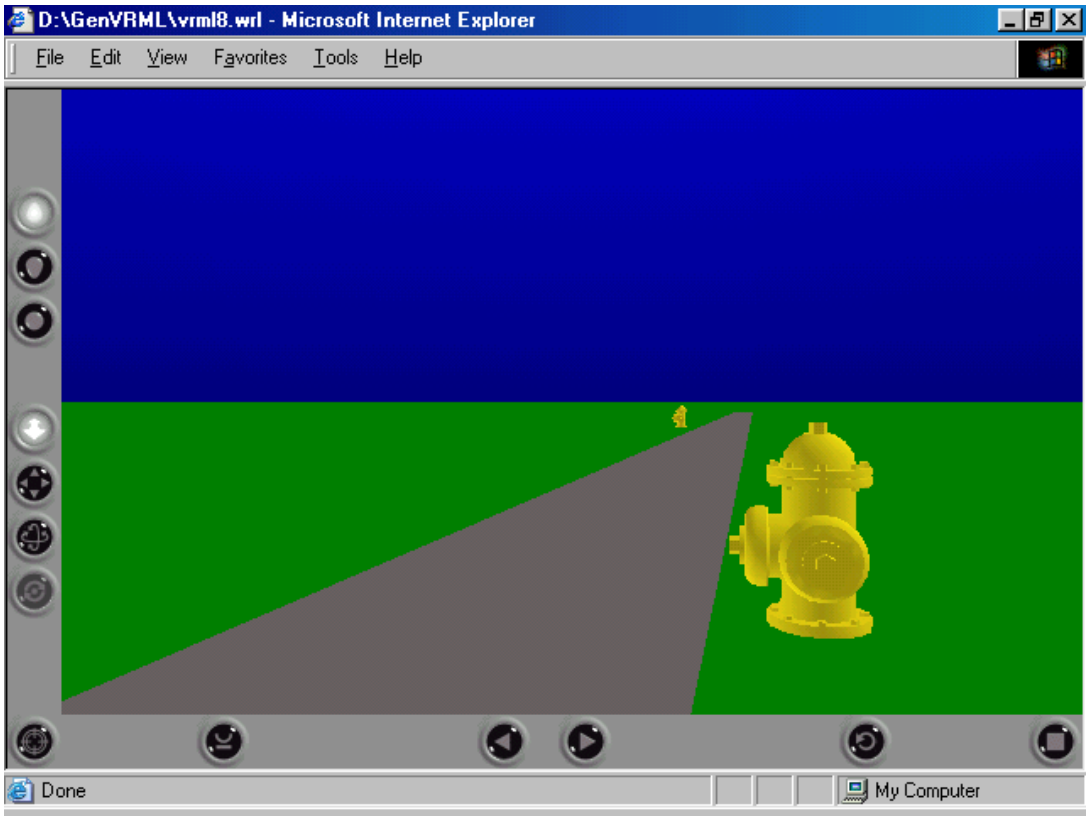

D:\GenVRML\vrml5.wrl - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Done
My Computer

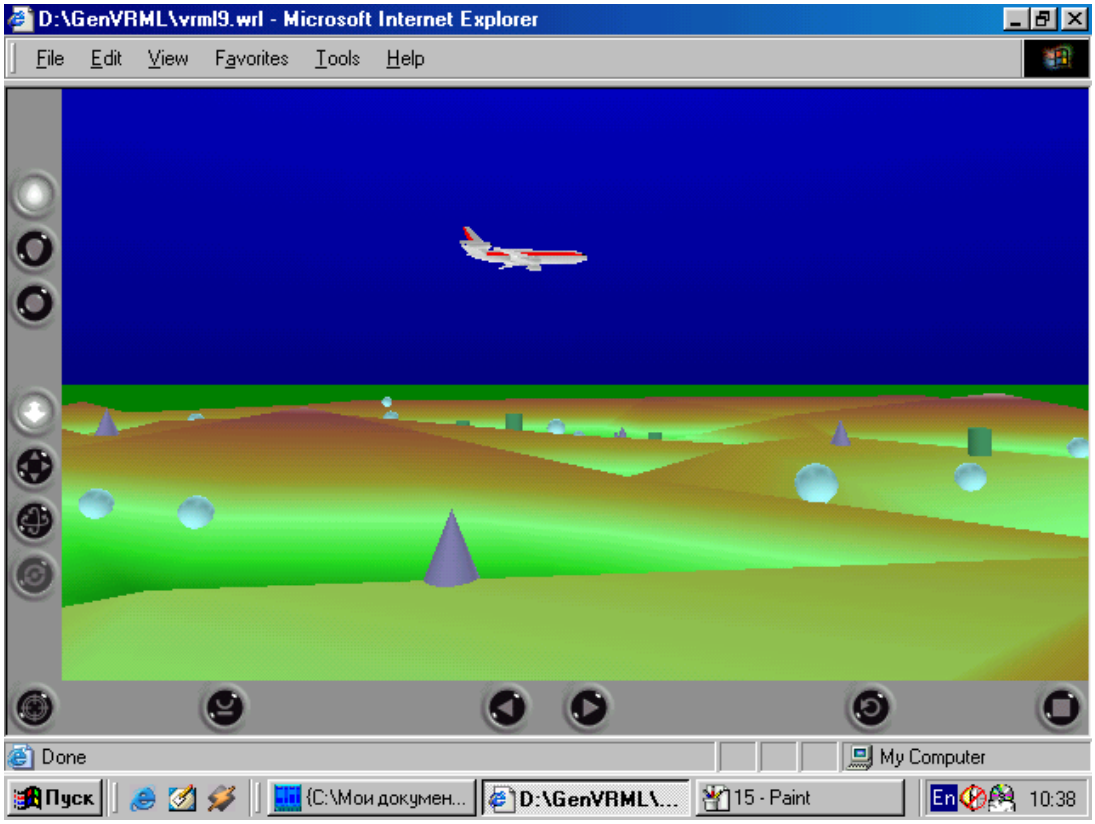

D:\GenVRML\vrml6.wrl - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Done
My Computer

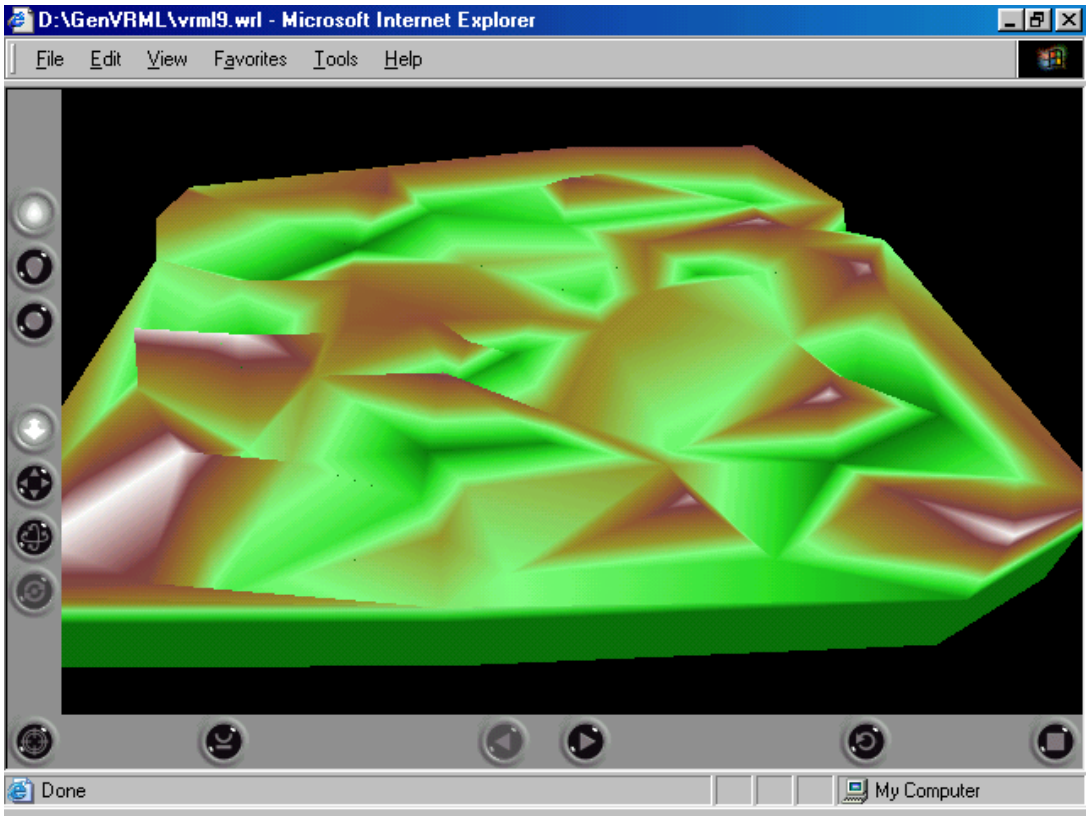

D:\GenVRML\vrml6.wrl - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Done
My Computer


D:\GenVRML\vrml6.wrl - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Done
My Computer


D:\GenVRML\vrml7.wrl - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Done
My Computer


D:\GenVRML\vrml8.wrl - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Done
My Computer


D:\GenVRML\vrml9.wrl - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Done
My Computer
Пуск
{C:\Мои докумен...
D:\GenVRML\...
15 - Paint
En
10:38


D:\GenVRML\vrml9.wrl - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Done
My Computer