Вокселизация Функциональных Форм

Катасонов А.В., Вяткин С.И., Долговесов Б.С.
Институт автоматики и электрометрии, СО РАН, Новосибирск, Россия
akatasonov@excelsior-usa.com, sivser@mail.ru, bsd@iae.nsk.su.

Аннотация

Перспективной областью использования вокселизации являются системы компактного представления воксельных массивов данных, в которых чрезвычайно экономичное функциональное представление на основе функций возмущения является форматом хранения данных. Целью данной работы стала разработка алгоритма и создание программных средств вокселизации функциональных форм.

Новизна данной работы заключается в адаптации уникального алгоритма визуализации свободных форм, основанного на рекурсивном разбиении пространства, к задаче вокселизации функциональных моделей.

Результаты данной работы позволяют визуализировать функционально заданные 3D сцены как на специальных воксельно-ориентированных видеоадаптерах, так и на любых других, используя софтверные библиотеки, такие как OpenGL, DirectX. Созданный программный модуль может использоваться как неотъемлемая часть систем компактного представления воксельных массивов данных.

***Ключевые слова: Вокселизация**, Рекурсивное Деление Объектного Пространства.*

1. ВВЕДЕНИЕ

Функциональное задание трехмерных сцен становится все более и более популярным в современной компьютерной графике. Наряду с полигональным и воксельным заданием, функциональное обладает рядом существенных преимуществ: небольшой размер хранимых данных и высокое качество изображений. Так же многие задачи 3D анимации можно решить только или, по крайней мере, наилучшим образом, используя аналитическое задание объектов. Например, такие анимационные эффекты как морфинг объектов, скиннинг (обтягивание скелета мышцами и кожей), морфинг с ограничениями (просачивание объектов через щели и т.д.) наилучшим способом реализуются в функциональном пространстве.

Существенным достоинством поверхностей свободных форм (базовые примитивы функционального пространства) является высокая степень гладкости. Чтобы достичь такой гладкости объектов в полигональном представлении используются такой метод как сплайн интерполяция, недостатком которой является использование большого количества полигонов для интерполяции даже небольших моделей. К тому же для сглаживания достаточно сложных полигональных объектов придется использовать большое количество сплайн патчей, что соответственно увеличивает время геометрической обработки сплайновой поверхности на величину прямо пропорциональную их количеству и поэтому полигональное представление в таком случае теряет свое основное достоинство – скорость геометрической обработки.

Скорость геометрической обработки является основным и возможно единственным недостатком функционального представления 3D сцен. Время просчета одного кадра может быть от нескольких секунд (при низком качестве визуализации и низком разрешении), до нескольких часов или десятков часов. В таком случае не может быть и речи об интерактивности процесса моделирования, что создает значительные проблемы при создании и редактировании объектов заданных аналитически.

Выходом может служить значительное снижение качества моделей на этапе моделирования, что создает большие неудобства и не дает требуемой интерактивности (не менее 30 кадров/сек.). Другим подходом может служить преобразование модели в другое представление, в котором будет возможность выполнять некоторые действия (поворачивать, перемещать, сжимать и т.д.) над объектами в реальном времени. Воксельное представление вполне подходит для этого, так как оно обладает высокой степенью точности представления данных и имеет аппаратные ускорители, обеспечивающие реальное время визуализации.

Воксельное представление является наиболее подходящим для большинства программных приложений в медицине, геологии, геодезии и т.д., где приходится работать с объемными снимками реального физического пространства. Снимки получаются путем использования различных сканеров и алгоритмов восстановления объема по 2D изображениям. Не исключено, что может потребоваться возможность создания синтезированных объемных снимков, путем внесения искусственно созданных моделей в воксельный объем. Учитывая перечисленные выше достоинства функционального представления, оно может служить средой моделирования искусственных объектов в такой ситуации.

Еще одной перспективной областью использования FV преобразования являются системы компактного представления воксельных массивов данных. Это системы ориентированные на существенное уменьшение количества памяти, занимаемое воксельным объемом, путем аппроксимации воксельных моделей аналитическими функциями. Обратное преобразование из функций в воксели (FV преобразование) также будет играть существенную роль в таких системах.

Подытоживая все выше сказанное, выделим основные причины, по которым задача FV преобразования становится достаточно важной и требующей решения:

- Возможность визуализировать функционально заданные 3D сцены на воксельно-ориентированных видео адаптерах с частотой не менее 30 кадров/сек.
- Возможность использования функциональных моделей в медицине, геологии и т.д., где основной

формат графических 3D данных – воксельный объем.

- Перспективной областью использования вокселизации являются системы компактного представления воксельных массивов данных, в которых чрезвычайно экономичное функциональное представление на основе функций возмущения является форматом хранения данных.

2. ИСТОРИЯ ВОПРОСА

Вокселизация это процесс преобразования геометрических объектов из их непрерывного геометрического представления в 3D массив элементов объема (воксели), которые наилучшим образом аппроксимируют непрерывный объект. Первые работы на данную тему основывались на алгоритмах построчного сканирования всего пространства для построения массивов вокселей [1, 2]. Это может быть представлено как расширение алгоритмов построчного 2D сканирования в двумерном случае для решения задачи растеризации (пикселизации) 2D геометрических объектов (см. Рис. 1) [3] на объемную область определения (см. Рис. 2). Данные алгоритмы называют алгоритмами построчного 3D сканирования. В двумерном случае получаемые пиксели напрямую визуализируются на экране, и к ним применяется фильтрация для снижения эффекта неровности (aliasing) изображения. Однако процесс вокселизации не предназначен для прямой визуализации объекта. Основной задачей является получение и сохранения дискретного объемного описания непрерывного объекта.

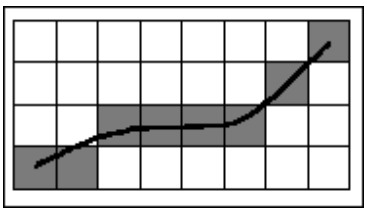

Рис. 1: Пример 2D кривой в дискретном случае (выделенные пикселы).

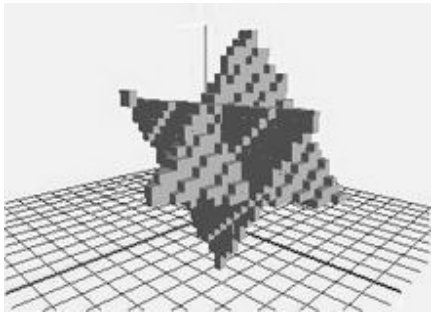

Рис. 2: Пример модели, образованной пересечением двух конусов в 3D дискретном представлении.

Бинарные технологии (binary techniques) были первыми алгоритмами вокселизации [1, 2]. В процессе точечной дискретизации значение непрерывного геометрического объекта вычисляется в центре вокселя. Размер 3D решетки определяет точность дискретного представления модели. Бинарные технологии вокселизации генерируют топологические и геометрически совместимые объекты, но они допускают эффект неровности (aliasing) объекта. Это особенно заметно при визуализации результирующих данных. Для решения данной проблемы были разработаны две технологии: технология фильтрации (filtering technique), пропускающая сгенерированный объект через специальный фильтр, сглаживающий форму объекта (antialiasing filter) [4, 5] и технология, основанная на полях расстояний (distance field technique), при которой каждому вокселю сцены приписывается расстояние до следующего объекта [6]. Для задач объемного анализа визуализация массивов вокселей не применяется, таким образом, визуальные дефекты, вносимые эффектом неровности (aliasing effect) не учитываются. Также, при анимации не обязательно иметь точный контур объектов, так как камера или точка обзора пользователя всегда находится на некотором расстоянии до объекта. Поэтому, для некоторых задач, бинарных технологий вокселизации оказывается достаточно. К сожалению, для комплексных моделей время вокселизации все еще далеко от того, чтобы быть интерактивным. Бинарные технологии могут быть использованы в любых приложениях, в которых процесс вокселизации может быть сделан заранее и не требует повторных запусков. Но когда приложение содержит динамические объекты, как например в динамических и интерактивных средах, модель должна быть вокселизированна повторно после того, как она была изменена. Это требует более быстрых решений для задачи вокселизации. Фэнг и Чен [7] использовали возможности аппаратных графических ускорителей для задачи вокселизации моделей конструктивной блочной геометрии (CSG models). Их алгоритм генерирует кусочки модели, используя процессор для работы с графикой поверхностей, для формирования конечного объемного представления. Это предоставляет достаточно быструю и интерактивную вокселизацию, в зависимости от графического процессора и размерности воксельного массива данных. Совершенно новым подходом к проблеме вокселизации явились работы Нило Столте [8, 9, 10]. Целью проводимых им работ было использование вокселизации для задачи визуализации неявных поверхностей (implicit surfaces). Неявные поверхности (implicit surfaces) определяются как функции вида F(x, y, z)=0. Интересным свойством таких поверхностей является возможность определения того, что точка находится внутри данной поверхности (когда $F(x, y, z) < 0$), вовне поверхности (когда $F(x, y, z) > 0$) и на поверхности (когда $F(x, y, z) = 0$). Хочется также отметить, что неявные поверхности являются основным и наиболее интересным способом задания функциональных форм. Но визуализация моделей созданных на базе неявных поверхностей является довольно ресурсоемкой задачей и требующей много времени. Ключевым моментом в процессе моделирования является именно время визуализации, которое должно предоставлять возможность интерактивной работы. Нило Столте добился заметных успехов в решении данной проблемы, используя вокселизацию как ключ к визуализации функциональных моделей. Перед тем как визуализировать модель, он предлагает предварительно её вокселизировать после чего визуализировать полученный массив вокселей, используя буфер глубины, поддерживаемый большинством видеоадаптеров.

Нило Столте в своих работах отмечает невозможность использования бинарных технологий для своих задач, поскольку они не гарантируют корректность модели и

требуют большого количества времени для вокселизации. Им были предложены новые технологии вокселизации гарантирующие корректность, таким образом, ни одна часть непрерывной функциональной поверхности не будет пропущена. Скорость работы предложенных им алгоритмов заметно выше, чем при использовании бинарных технологий. Также преимуществом его методов является простота распараллеливания процесса вокселизации [11].

Алгоритмы, предложенные Нило Столте, относятся к классу алгоритмов с рекурсивным разбиением пространства сцены. Базой в предложенных алгоритмах является интервальная арифметика. Интервальная арифметика была впервые использована для решения задач компьютерной графики в работах Шнайдера и Даффа [12, 13]. Несмотря на то, что ими были представлены многие полезные свойства интервальной арифметики, она все еще не находит широкого применения для решения задач компьютерной графики. Нило Столте заметил, что интервалы, в частности, полезны для определения дискретных моделей [8, 9, 10], так как любой трехмерный куб в пространстве может быть формально представлен тремя интервалами, каждый из которых соответствует одной из трех осей координат. Вдобавок, если непрерывная модель задана, неявными поверхностями, то интервальная арифметика может быть использована для определения того, содержит ли трехмерный куб непрерывную модель внутри себя или нет.

Интервальная арифметика гарантирует, что результат любой арифметической операции находится между двумя значениями, называемыми *границами интервалов*. Любое действительное число представляется двумя границами интервала. Например, координаты X, Y и Z в интервальной арифметике представляются в следующем виде: **X** = [x; X], **Y** = [y; Y], **Z** = [z; Z].

В алгоритмах, предложенных Нило Столте, границы данных интервалов являются координатами границ октанта (куба). Замещая, в вычислениях неявной функции каждую регулярную координату, соответствующим интервалом и каждую регулярную операцию соответствующей интервальной операцией, мы получим интервальный вариант данной функции, которую Шнайдер [12] называет *включающей функцией* (*inclusion function*). Далее мы можем определить, что поверхность не проходит через заданный октант, проверяя, что полученный интервал не *включает* нуль. Таким образом, определяется, что результирующий интервал *включающей функции* не включает решения для регулярной функции $F(x, y, z) = 0$. Тогда если результирующий интервал не включает нуль, функция соответственно не имеет нуля в заданном октанте; таким образом, поверхность не проходит через октант.

Вычисления в таком случае более рациональны, чем в алгоритмах построчного сканирования, требующих вычисления пересечений с моделью. И что не менее важно, гарантируется корректность результата вычисления. Однако, интервальная арифметика очень консервативна, что означает, что использование больших трехмерных кубов также вызывает больше переоценки. Предложенный Нило Столте способ рекурсивного разбиения пространства [8, 9, 10] является элегантным способом решения данной проблемы. На каждом шаге объем разбивается на восемь частей, которые рассматриваются уже отдельно, таким образом, происходит очень быстрое приближение к поверхности модели. Вокселизация завершается, когда рекурсивное разбиение достигает необходимой размерности трехмерного массива вокселей. К тому же, данная процедура вокселизации не требует строго заданной размерности массива. Вдобавок, процесс разбиения может быть остановлен в любое время и продолжен с прежнего состояния. Это является одним из самых значительных преимуществ, отличающих данный метод от методов разработанных ранее.

В соотношении с интервальной арифметикой неявные поверхности не только мощный инструмент моделирования, но может быть основным рабочим инструментом в контексте дискретного представления. Становится возможным определять воксели внутри, снаружи или на поверхности, заданной неявной функции. В этом случае метод, разработанный Нило Столте, имеет еще одно важное преимущество: он позволяет определять воксели не только на поверхности модели, но также и внутри её.

3. ВОКСЕЛЬНОЕ ПРЕДСТАВЛЕНИЕ ДАННЫХ

Множество объемных данных это трех мерный массив значений ассоциированных с точками в трех мерном пространстве. В большинстве случаев, множеством объемных данных представляется некоторый физический объект или явление в некотором физическом пространстве. Иногда такие объекты еще называют *объемными объектами*. Каждый элемент в трехмерном массиве называется объемным элементом или вокселем (по аналогии с пикселем, который обозначает элемент двумерного изображения). Множество трехмерных целочисленных координат (x, y, z) обозначает позицию вокселя внутри 3D массива.

Значение данных в вокселе – это количественное представление одной или более характеристик объекта или явления в небольшой области 3D пространства окружающего позицию центра данного вокселя. Такими характеристиками могут быть: плотность, температура, скорость, ускорение, материал, цвет, прозрачность или что-то другое в зависимости от конкретного приложения.

Наиболее важными структурами для описания объемных данных являются:

- бинарная воксельная модель: воксели могут принимать два значения: 1 (объект) или 0 (нет объекта). Эта очень простая модель и используется редко. Для того, чтобы уменьшить необходимый для хранения объем памяти, бинарные объемы могут быть рекурсивно разбиты на меньшие объемы, содержащие воксели равной величины; результирующая структура данных называется 8-деревом или октантным деревом.

- полутоновая воксельная модель: каждый воксель содержит информацию об интенсивности. Для полутоновых объемов также разработаны структуры в виде октантного дерева [19].

- обобщенная воксельная модель: кроме информации об интенсивности каждый воксель содержит атрибуты, характеризующие его принадлежность к различным объектам, и/или данные от других источников (например, MRI или PET).

- "интеллектуальные объемы": в качестве развития обобщенной воксельной модели рассматривается модель, в которой свойства объектов (такие как цвет, имена в различных языках, указатели на дополнительную

информацию) и их взаимосвязи моделируются на символьном уровне. Подобная структура данных является основной для таких продвинутых приложений, как медицинские атласы.

4. ВОКСЕЛИЗАЦИЯ МОДЕЛЕЙ НА БАЗЕ ФУНКЦИЙ ВОЗМУЩЕНИЯ

В данной работе использовался алгоритм многоуровневого отслеживания лучей [15] для визуализации моделей на базе функций возмущения [16], адаптированный для задачи вокселизации. Функционально заданные геометрические объекты с применением функций возмущения достаточно подробно изложены в работах [15-17].

На первом шаге рекурсии куб, который ограничивает функциональную модель, разбивается на четыре меньших параллелепипеда (бруска) в экранной плоскости (по аналогии с пирамидой видимости [15]). На этапе деления пространства по четверичному дереву, где совершается сжатие в два раза и перенос на ±1 по двум координатам, происходит рекурсивное преобразование коэффициентов базовых поверхностей и функций возмущения. Полученные коэффициенты используются в тесте на пересечение. Для каждого нового бруска выполняется тест на пересечение с объектом. На основании результатов этого теста осуществляется деление брусков, которые лежат внутри объекта целиком или, возможно, частично, а заведомо внешние бруски исключаются из обработки. Деление брусков по четверичному дереву ведется до тех пор, пока не достигается максимально установленный уровень рекурсии. Далее осуществляется бинарное деление бруска по Z-координате (назовем его вокселем верхнего уровня). Воксель на уровне i разбивается на две равные части (на два вокселя следующего уровня детальности см. Рис. 3). Далее, каждая часть рекурсивно разбивается в поиске вокселей, принадлежащих поверхности. Если какой-то воксель не принадлежит поверхности, то он далее не рассматривается (см. Рис. 4). Разбиение продолжается до тех пор, пока не будет достигнут заданный уровень детальности.

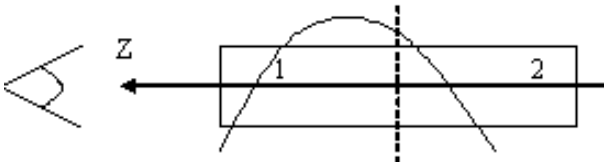

Рис. 3: В процессе рекурсивного подделения, на текущем уровне детальности воксель разбивается вдоль оси Z на два вокселя следующего уровня детальности (помечены как 1 и 2). Воксели 1 и 2 проверяются на пересечение с поверхностью, и если один из вокселей оказывается пустой, то он выбрасывается из рассмотрения.

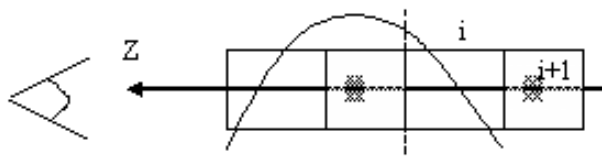

Рис. 4: На i+1 уровне рекурсии отмеченные воксели далее не участвуют в разбиении, так как они не содержат поверхности внутри себя.

Для визуализации свободных форм достаточно рассматривать только воксели ближние к наблюдателю, если они содержат поверхность (на Рис. 3 это воксель 1). Но для вокселизации нам необходимо найти все воксели поверхности, поэтому рассматриваются все воксели следующего уровня детальности, содержащие поверхность.

На этом этапе для текущего уровня рекурсии вектор конца вокселя, являющегося ближайшим по отношению к наблюдателю, полагается равным вектору ближайшего конца вокселя предыдущего уровня деления. Вектор дальнего конца вокселя вычисляется как полусумма векторов ближнего и дальнего концов вокселя предыдущего уровня деления. Для дальнего, по отношению к наблюдателю, конца вокселя вектор ближайшего конца равен полусумме векторов ближнего и дальнего концов вокселя предыдущего уровня деления, а вектор дальнего конца вокселя равен вектору дальнего конца вокселя предыдущего уровня деления.

$P_{ni}^{1} = P_{ni-1}$, $P_{fi}^{1} = (P_{ni-1} + P_{fi-1})/2$.

$P_{ni}^{2} = (P_{ni-1} + P_{fi-1})/2$, $P_{fi}^{2} = P_{fi-1}$.

$V_i^1 = \{P_{ni}^1, P_{fi}^1\}$, $V_i^2 = \{P_{ni}^2, P_{fi}^2\}$, где V_i^1 -воксель i-го уровня рекурсии, ближайший к точке наблюдения, V_i^2 -воксель i-го уровня рекурсии, дальний от точки наблюдения. P_{ni}^k и P_{fi}^k - координаты ближнего по отношению к наблюдателю, и дальнего концов вокселя k (где k из {1, 2}) на i-м уровне рекурсии.

При визуализации и/или вокселизации поверхностей проверяется тест на принадлежность им лишь пересеченных вокселей, внешние и внутренние воксели отбраковываются. Для того, чтобы повысить реалистичность изображения и расширить класс отображаемых объектов (полупрозрачные структуры с внутренним распределением плотности, 3D текстуры), необходимо отображать внутреннюю полупрозрачную структуру объекта. Для этого в формировании изображения должны участвовать не только воксели, которые лежат на поверхности, а так же и те, которые находятся внутри объекта. Следовательно, при делении объема внутренние части объекта не отбрасываются, для них проводится дальнейшая рекурсия алгоритма. При сканировании сцены по Z-координате, что соответствует сканированию объема в глубину, сканирование не прерывается при встрече с поверхностью, а работает далее, пока полностью не просканируется объем или не накопится определенное значение прозрачности, большее некоторого порогового значения. После определения вокселя последнего уровня разбиения, содержащего поверхность, информацию о цвете и нормали в соответствующей точке поверхности добавляются в результирующий массив воксельных данных. Описания соответствующего алгоритма вычисления цвета в точке поверхности можно найти в работах [15, 16]. Координаты точки в 3D массиве вокселей вычисляются по следующим формулам:

$$i = (x * N)/2$$
$$j = (y * N)/2$$
$$k = (z * N)/2$$

где i, j, k целочисленные позиции в массиве вокселей соответствующие координатным осям X, Y, Z, (x, y, z) – вещественные координаты точки в видимом объеме, N –

разрешение массива вокселей. Видимый объем в системе VxDemo имеет границы [-1, 1].

5. ЗАКЛЮЧЕНИЕ

Результатом данной работы является программный модуль, выполняющий преобразование функционально заданной 3D сцены в воксельное представление. Данный модуль встроен в систему визуализации функциональных форм VxDemo и успешно применяется для экспортирования смоделированной 3D сцены в воксельное представление. На текущий момент система поддерживает три формата экспорта: BINVOX, MIRA, TRvox1999a. Последний тип описания воксельных данных применяется в видеоадаптерах компании TeraRecon. Форматы BINVOX и MIRA являются широко распространенными в различных системах визуализации научных данных, представленных в виде массивов вокселей. Полученные данные могут также визуализироваться свободно распространяемым программным продуктом ViewVox разработанным в Universiteit Utrecht. Модуль вокселизации функциональных форм написан на языке C++ с использованием библиотек STL и BOOST. Средой разработки был выбран продукт Microsoft Visual Studio 2003.

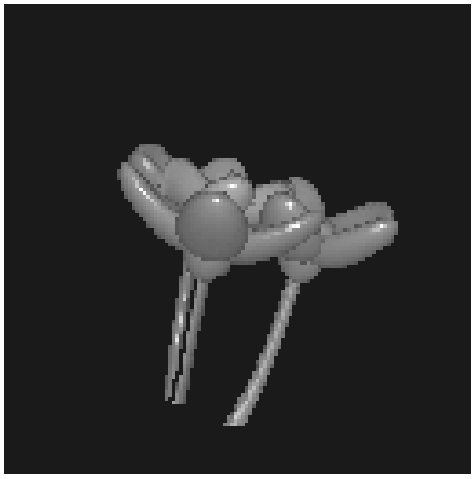

Рис. 5: Функциональная модель смоделированная в системе VxDemo. В состав модели входят около 25 функций. Модель визуализирована при разрешении 128 x 128 точек. Глубина рекурсии четверичного разбиения равна 7, глубина рекурсии бинарного разбиения равна 7.

С помощью разработанного программного продукта, дизайнер, моделирующий 3D модели в системе VxDemo, может интерактивно производить манипуляции над создаваемой моделью (поворачивать, масштабировать, перемещать и т.д.) предварительно вокселизировав её. Полученные модели, теперь могут использоваться в других системах визуализации и обработки графических данных (например, в системах визуализации медицинских данных), что существенно расширяет область применения и общую ценность системы VxDemo. Пользователь может контролировать скорость вокселизации и качество получаемой воксельной модели, манипулируя параметрами Qr, Br (глубина рекурсии при четверичном разбиении пространства в плоскости XY и глубина рекурсии при бинарном разбиении вдоль оси Z). Также пользователь может повышать или понижать детальность отдельных частей модели или, вообще, вокселизировать только отдельные её части. При экспорте модели в воксели, наряду с пространственными координатами точек поверхности, может сохраняться также информация о цвете в данной точке поверхности (см. Рис. 7).

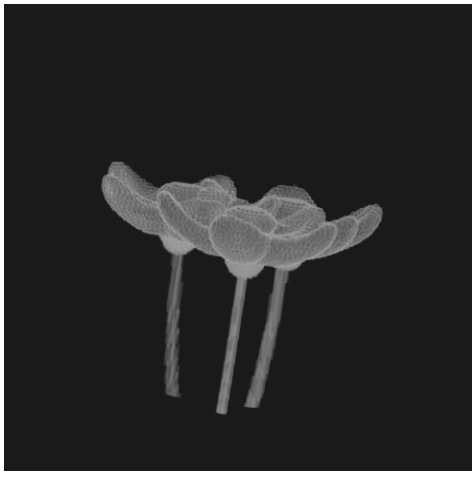

Рис. 6: Воксельное представление модели смоделированной в системе VxDemo и вокселизированной с помощью разработанного модуля вокселизации. Глубина рекурсии четверичного разбиения равна 7, глубина рекурсии бинарного разбиения равна 7. Разрешение получившегося массива вокселей равна 128x128x128. Время вокселизации составило 3 секунды. Модель визуализирована на графическом адаптере Volume Pro 1000 от компании TeraRecon Inc.

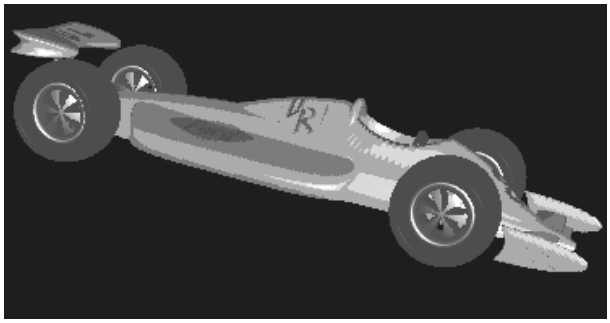

Рис. 7: Вокселизированная модель смоделированная в системе VxDemo. В состав модели входит 42 функций. Глубина рекурсии четверичного разбиения равна 9, глубина рекурсии бинарного разбиения равна 9. Разрешение получившегося массива вокселей равна 512x512x512. Время вокселизации составило 1.5 минуты. Модель визуализирована в программе ViewVox.

6. REFERENCES

[1] Kaufman, A. and Shimony, E., '3D Scan-Conversion Algorithms for Voxel-Based Graphics', Proc. ACM Workshop on Interactive 3D Graphics, Chapel Hill, NC, October 1986, pp. 45-76.

[2] Kaufman, A., "Efficient Algorithms for 3D Scan-Conversion of Parametric Curves, Surfaces, and Volumes", Computer Graphics, 21, 4 (July 1987), pp. 171-179.
[3] James D. Foley, Andries van Dam, Steven F. Feiner, and John F. Hughes. *Computer Graphics: Principles and Practise*. Addison-Wesley Systems Programming Series. Addison-Wesley Publishing Company, 2nd edition, 1990.
[4] Milos Sramek and Arie E. Kaufman. Alias-free voxelization of geometric objects. *IEEE Transactions on Visualization and Computer Graphics*, 5(3):251–267, July 1999.
[5] S.Wang and A. Kaufman. Volume sampled voxelization of geometric primitives. In *Proceedings of Visualization' 93*, pages 78–84. IEEE, 1993.
[6] Sarah F. Frisken, Ronald N. Perry, Alyn P. Rockwood, and Thouis R. Jones. Adaptively sampled distance fields: A general representation of shape for computer graphics. *Proceedings of SIGGRAPH 2000*, pages 249–254, July 2000. ISBN 1-58113-208-5.
[7] Shiaofen Fang and Hongsheng Chen. Hardware accelerated voxelization. *Computers and Graphics*, 24(3):433–442, June 2000.
[8] Nilo Stolte and Arie Kaufman. Novel Techniques for Robust Voxelization and Visualization of Implicit Surfaces. Graphical Models, 63(6):387-412, Academic Press, 2001.
[9] Nilo Stolte and Arie Kaufman. Robust Hierarchical Voxel Models for Representation and Interactive Visualization for Implicit Surfaces in Spherical Coordinates. In *Implicit Surfaces'98*, pages 11-17, Seattle, 1998.
[10] Nilo Stolte. Robust Voxelization of Surfaces. Technical Report TR.97.06.23, Center for Visual Computing, State University of New York at Stony Brook, 1997.
[11] Nilo Stolte and Arie Kaufman. Parallel Spatial Enumeration of Implicit Surfaces using Interval Arithmetic for Octree Generation and its direct Visualization. In *Implicit Surfaces'98*, pages 81-87, Seattle, 1998.
[12] John M. Snyder. Interval Analysis For Computer Graphics. Computer Graphics, 26(2):121-130, July 1992.
[13] Tom Duff. Interval Arithmetic and Recursive Subdivision for Implicit Functions and Constructive Solid Geometry. Computer Graphics, 26(2):131{138, July 1992.
[14] Laur, D., Hanrahan, P.: Hierarchical splatting: A progressive refinement algorithm for volume rendering. Comput. Graphics 25, 4 (1991), pp. 285-288.
[15] http://www.cgg.ru/febr/vjat/pivweb.html#a
[16] Sergei I. Vyatkin, Boris S. Dolgovesov, Oleg Y. Guimaoutdinov - "Synthesis of Virtual Environment Using Perturbation Functions", volume III (Emergent Computing and Virtual Engineering), World Multiconference on Systemics, Cybernetics and Informatics Proceedings, Orlando, FL, USA, July 22-25, 2001, P.350.
[17]http://www.graphicon.ru/2002/pdf/Vyatkin_Dolgovesov_Re.pdf

Voxelization Of Functional Forms

The voxelization technique subdivides the object space in a recursive way testing using modified multilevel ray casting algorithm, if a given bar and voxel contains a part of the free form surface based on perturbation functions. When the voxel level arrives, the normal is calculated by the gradient in the middle of the voxel. The main advantage of our representation is it robustness. This representation can be locally refined to any desired resolution from any previous voxelized part of the surface. These qualities alone make it a much better way of representing curved complex surfaces in comparison with polygonal representation.

About the authors

Alexander Katasonov is a student at Novosibirsk State University, Department of Information Technologies. His contact email is akatasonov@excelsior-usa.com

Sergei I. Vyatkin (Ph.D.) is a scientific researcher of Synthesizing Visualization Systems Laboratory at Institute of Automation and Electrometry, SB RAS.
His contact email is sivser@mail.ru

Boris S. Dolgovesov (Ph.D.) is a head of Synthesizing Visualization Systems Laboratory at Institute of Automation and Electrometry, SB RAS.
His contact email is bsd@iae.nsk.su