Наукові праці ДонНТУ
Серія "Інформатика, кібернетика та обчислювальна техніка" *випуск 12(165), 2010*

УДК 004.932

Сжатие графической информации с использованием тематических текстур

Вяткин С.И.[1], Романюк О.В.[2]
[1]Институт автоматики и электрометрии СО РАН, Новосибирск, Россия,
[2] Винницкий национальный технический университет, Винница, Украина,
sivser@mail.ru

Abstract

***Vyatkin S.I., Romanyuk O.V. Graphical information compression by thematic textures using**. To decorate surface of terrain thematic textures are used. It is supposed that observer is not often interested in the exact photographic information about the wide areas covered, for instance, by forest, water, etc., so that the benefits of photographic texture can be obtained by composing a number of patterns called "themes" to produce a final texture of the specific area. The goal of using thematic textures is to abolish the need of the global area texture, and additionally, animation features become available by simple means.*

Введение

Компьютерная графика реального времени, ориентированная на визуализацию трехмерных сцен, достигла на сегодняшний день существенных успехов. Она находит широкое применение от сложных систем визуализации для тренажерных комплексов (авиационных, космических, морских, автомобильных и т.п.) до графических акселераторов, используемых в компьютерных играх. И хотя достигнут достаточно высокий реализм отображаемых сцен в реальном времени, существует ряд задач, например, отображение больших районов местности, где требуется хранить и отображать сцены, содержащие большое количество графической информации.

Повышение реализма достигается уже не только за счет количества примитивов, которыми представлена сцена, а за счет моделирования специальных эффектов окружающей среды и мелких деталей поверхности объектов сцены. Алгоритмы построения изображения, применяемые в современных графических акселераторах, не решают эту задачу в полном объеме.

Для реалистичности изображаемой местности требуется текстура большого размера. Обычно для этого используется фотография той реальной местности, которая предполагается использоваться в системе визуализации.

Поэтому необходимо сжатие не только геометрических данных, т.е. примитивов, но и текстуры [1]. В настоящее время в компьютерной графике используется несколько способов представления геометрических объектов, каждый из которых, в силу своих свойств, используется в различных областях: от систем 3D - моделирования и CAD-систем до систем визуализации реального времени. Полигональное задание является кусочно-линейной аппроксимацией некой поверхности. Достоинства такого задания заключаются в простоте представления и универсальности (кусочно-линейное представление существует для любой поверхности). Недостатки - это большие размеры баз данных, необходимые для хранения информации о геометрии поверхности. К настоящему времени разработано довольно много алгоритмов, предназначенных для построения адаптивной триангуляции рельефа путем использования иерархических структур данных (бинарных деревьев, квадродеревьев) [2-4]. Основное назначение таких алгоритмов – сократить число выводимых треугольников без существенной потери качества изображения. Однако для сложных поверхностей, каким является, к примеру, горный рельеф местности, требуется огромное число треугольников. Функциональное представление [5] наиболее точно из всех существующих описывает геометрию объекта и имеет наименьший размер данных, необходимых для описания геометрии объекта. В работе [6] приведены результаты исследований по моделированию и визуализации сложных поверхностей с применением скалярных функций возмущения. Анализ возможных путей развития графических систем и, в частности, систем визуализации реального времени, показывает, что самый простой способ достижения реалистичности изображения – за счет увеличения количества отображаемых в кадре многоугольников - не самый эффективный. На этом пути трудно ожидать качественного скачка, например, такого, какой дало в свое время введение текстуры. Даже при многократно большем количестве отображаемых многоугольников изображение без текстуры будет беднее, чем более простое, но с цветной текстурой.

В данной работе предлагается метод сжатия текстурных данных больших районов местности.

Постановка задачи

Для текстурирования поверхности земли используется тематическая текстура, еще ее называют композитной. Методика исходит из предположения о том, что наблюдателю не надо точной (фотографической) визуальной информации в областях, занимаемых, например, лесом, морем и т.д. Достаточно заполнить такие области обобщенными текстурными образцами - "темами". Области фототекстуры местности с одной темой ограничиваются кривыми. Каждой области приписан указатель на текстурную тему и процедуру генерации границ. Это требует дополнительного канала обработки для преобразования входных данных из памяти базы данных в текстурную карту нужного разрешения. Однако тогда можно не иметь глобальной текстуры земной поверхности, суммарный объем памяти, которой в несколько раз больше. Кроме этого появляются новые возможности анимации текстуры.

Перечислим несколько условий, необходимых для того, чтобы получить сжатие корректным образом.

Во-первых, предположим, что изображение местности состоит из однородных областей, каждая из которых может быть представлена своей собственной текстурой. Например, лес может быть представлен текстурой “лес”, поле – текстурой “поле”, и т.д. Тогда результирующее изображение можно представить в виде тематической текстуры, состоящей из массива индексов, являющихся определённым номером темы [1]. Каждая тема – это небольшая однородная текстура (текстура леса, поля, водной поверхности и т.д.), удовлетворяющая условию сшивки – левая граница совпадает с правой соседней областью, верхняя – с нижней.

Во-вторых, в задаче отображения местности нам не важно точное совпадение исходной фототекстуры и сжатой тематической текстуры. Сжатие исходного изображения происходит за счёт его преобразования в тематическую текстуру с потерей информации (аналогия - JPEG).

Так как текстура должна находиться в постоянно сжатом состоянии на этапе непосредственного применения в системе визуализации, стандартные методы сжатия неприменимы, потому что требуют дополнительных вычислений при распаковке и дополнительную память системы.

Описание метода

В предлагаемом методе цвет изображения в определённой точке вычисляется в процессе визуализации. Результирующее сжатие осуществляется следующим способом. Как было выше сказано, тематическая текстура представляет набор индексов. Обычная текстура состоит из текселей, хранящих значение цвета в каком-либо формате (индекс в палитре цветов или полное RGB-значение). Индексы тематической текстуры представляют собой сетку. Каждая ячейка этой сетки, образуемая четырьмя соседними индексами, кодирует N*M текселей, где N и M параметры, задающие размер ячейки. Поэтому, результирующее сжатие пропорционально произведению M на N. Чем больше эти параметры, тем больше степень сжатия, что в свою очередь, приводит к снижению детализации изображения тематической текстуры. Поэтому эти параметры подбираются вручную так, чтобы получить правильный визуальный эффект.

При отображении тематической текстуры, каждая ячейка обрабатывается отдельно. Метод вычисления состоит из нескольких шагов:

1. Проверяем, если все четыре соседних индекса рассматриваемой ячейки тематической текстуры совпадают, то изображение данного участка образуется одной темой.
2. Иначе, вычисляем цвет каждой точки данного участка изображения путём замешивания цветов тем, определяемых индексами текущей ячейки тематической текстуры.

Замешивание в п.2 необходимо для того, чтобы обеспечить плавность перехода на границе между различными темами. Результирующий цвет получается равным взвешенной сумме цветов трёх соседних тем, где вес определяется расстоянием между текущей точкой ячейки до вершины ячейки. Для этого прямоугольная ячейка разбивается на два треугольника (рис 1.).

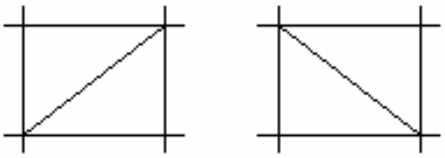

Рисунок 1 - Способы разбиения ячейки на треугольники

Выбор способа разбиения ячейки необходим для того, чтобы не возникали дефекты при отображении. В данном методе предлагается производить разбиение по диагонали ячейки, концы которой соответствуют одинаковым индексам. Если все индексы различны, то разбиваем произвольным способом. На рис. 2 изображён участок тематической текстуры. При

её построении в качестве массива индексов использовалось обычное изображение (рис. 3), предварительно обработанное в редакторе. Выделенная прямоугольником область, изображенная на рис. 3, соответствует изображению тематической текстуры на рис. 2. Каждому индексу соответствует свой цвет, определённый пользователем.

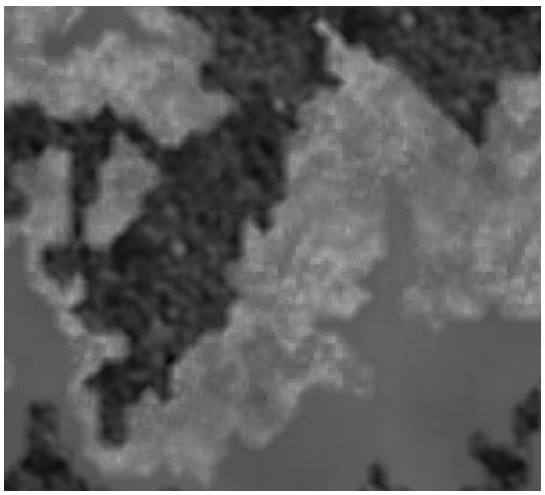

Рисунок 2 - Пример тематической текстуры

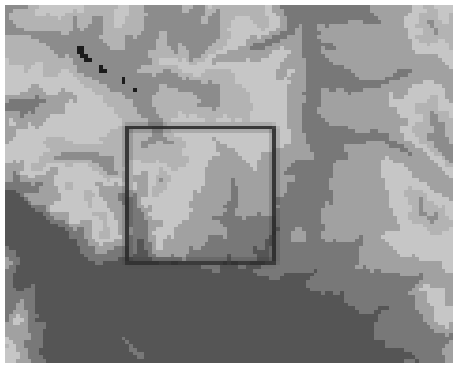

Рисунок 3 - Участок изображения карты индексов

На рисунке 4 показан алгоритм вычисления цвета в прямоугольной области R. Пусть T [N] – массив из N тематических текстур (тем), DAT – карта индексов DAT (i, j), где индекс определяет номер темы. Результирующее изображение получается в результате вычисления цвета в каждой прямоугольной области R [i][j], попадающей в область запроса.

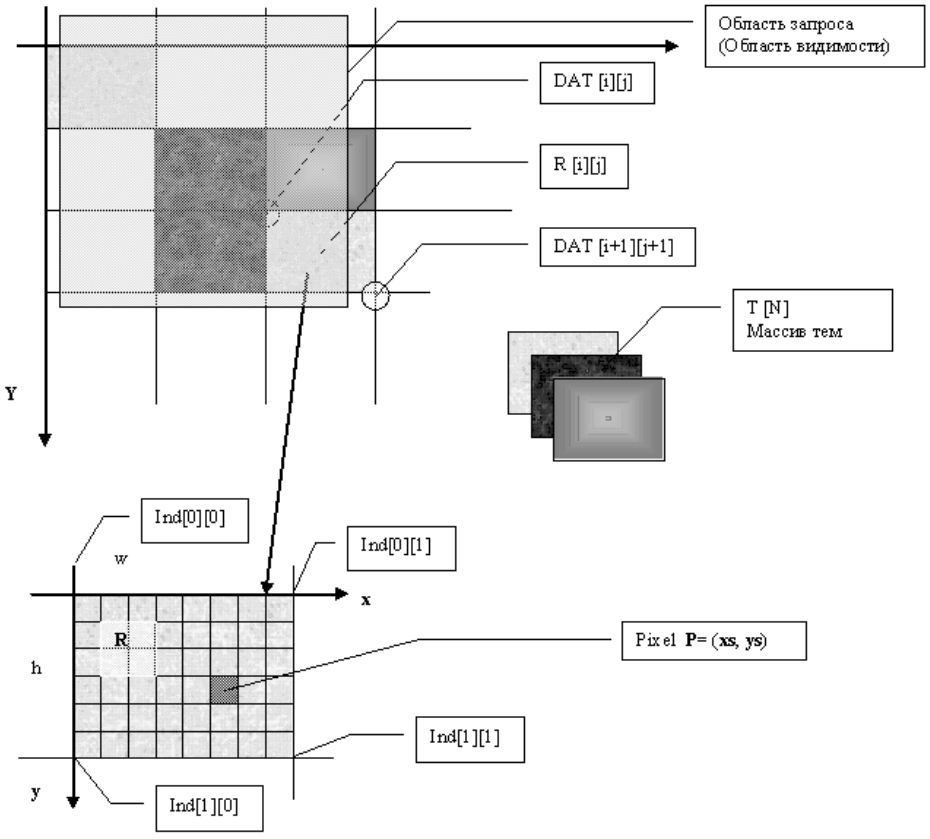

Рисунок 4 - Вычисления цвета в прямоугольной области

Алгоритм вычисления цвета в прямоугольной области **R** следующий.

MAP – экран (область видимости)
FLOAT **x**, **y** –координаты в системе прямоугольника **R** (**0 <= x <= 1, 0 <= y <= 1**).
INTEGER **x0**, **y0** глобальные координаты левого верхнего угла прямоугольника **R**
INTEGER **xs0**, **ys0** –экранные координаты левого верхнего угла прямоугольника **R**

Ind[2][2] – массив 2 на 2 соседних индексов прямоугольника **R**
Ind= { DAT(i, j), DAT(i+1, j) DAT(i, j+1), DAT(i+1, j+1)}

```
If(DAT(i,j)==DAT(i+1,j)==DAT(i+1,j+1)==DAT(i,j+1))
{
        Fill R by thema T[ DAT(i,j) ]
}
else if  ( DAT(i, j) == DAT (i + 1, j + 1) )
{
        dy= 1/h
        dx= 1/w
        for (y= 0, ys= 0; ys<h; ys++, y+= dy)
{
        for (x= 0, xs= 0; xs<w; xs++, x+= dx)
{
if ( x <= y )
        MAP(xs + xs0, ys + ys0)=

        T(ind[1][0], xs + x0, ys + y0)*(y - x) +

        T(ind[0][0], xs + x0, ys + y0)*(1 - y) +

        T(ind[1][1], xs + x0, ys + y0)*x ;
                                        else
        MAP(xs + xs0, ys + ys0)=

        T(ind[0][1], xs + x0, ys + y0)*(x - y) +

        T(ind[0][0], xs + x0, ys + y0)*(1 - x) +

        T(ind[1][1], xs + x0, ys + y0)*y ;
                                }
                }
        else
        {
                dy= 1/h
                dx= 1/w
        for (y= 0, ys= 0; ys<h; ys++, y+= dy)
                {
for (x= 0, xs= 0; xs<w; xs++, x+= dx)
                                {
                if ( x+y <= 1 )
        MAP(xs + xs0, ys + ys0)=

        T(ind[0][0], xs + x0, ys + y0)*(1 - x - y) +

        T(ind[0][1], xs + x0, ys + y0)*x +

        T(ind[1][0], xs + x0, ys + y0)*y
                else
        MAP(xs + xs0, ys + ys0)=

        T(ind[1][1], xs + x0, ys + y0)*(x + y - 1) +

        T(ind[0][1], xs + x0, ys + y0)*(1 - y) +

        T(ind[1][0], xs + x0, ys + y0)*(1 - x) ;
                                }
                }
}
```

Заключение

Разработан метод создания и отображения тематической текстуры, позволяющий значительно снизить затраты на память системы визуализации. Что касается дальнейших планов, то предполагается разработать способы интерактивного взаимодействия с построенной сценой. В частности, предполагается разработка алгоритмов изменения текстуры пользователем во время растрирования сцены. То есть необходимо создать некий удобный аппарат визуального воздействия на сцену.

Имеются несколько основных областей применения результатов данной работы, это, в первую очередь, тренажеры (космические, авиационные и т.д.), которые требуют эффективной работы с большой базой данных, описывающей моделируемую обстановку в реальном времени. Алгоритмы корректно решающие задачи данного применения, можно определить как алгоритмы визуализации незамкнутой базы данных. Под незамкнутостью базы данных понимается то, что база данных не входит целиком в поле зрения. Перспективные системы вооружений и военной техники, предъявляют принципиально новые требования к компьютерным системам визуализации (КСВ), используемым в тренажерах для генерации изображения окружающей обстановки. К наиболее сложным можно отнести задачи, связанные с полетом на малых высотах с огибанием рельефа местности, командными учениями, танковые, вертолетные и ряд других. Создание перспективных КСВ требует решения таких проблем как:

- Повышение реализма отображаемых сцен как за счет существенного повышения производительности систем (на порядок и более), так и за счет воспроизведения различных визуальных эффектов (различного вида текстура, газ, дым, дождь, снег, объемные облака и др.).

- Отображение больших районов местности (более 400х400 км).
- Отображение обстановки на большие экраны с углом обзора до 360 град. при подвижном наблюдателе и проекторе. Проецирование изображения на сферический экран большого размера (купол) - актуальная сложная задача, решение которой необходимо главным образом для тренажеров военного применения. Сложность и объем вычислений существенно увеличиваются при учете подвижности наблюдателя и проектора.
- Отображение рельефа местности по карте высот. Известно, что генерируют ландшафт, аппроксимируя его многоугольниками на сетках высот. Однако вычислительная мощность современных систем визуализации реального времени еще не превысила рубежа 100000 многоугольников за кадр (60 Гц), с пиксельной производительностью - 2,1 млн. пикселей в одном канале. В то время как для отображения реалистичного горного рельефа требуется большая производительность.
- Анимация и деформация объектов и поверхностей. Возможность описания динамических объектов типа волн, движения поверхности, позволяет визуализировать движение волн, деформацию местности (взрывы, воронки в земле, фонтаны воды во время морского боя, погружение субмарины в волнующееся море с учетом полупрозрачности воды, изменение форм облаков и т.д.).

Решение перечисленных проблем традиционными способами, используемыми в зарубежных и отечественных системах, выпускаемых ведущими фирмами, сопряжено с существенным усложнением оборудования и стоимости систем.

Литература

1. Sergei I. Vyatkin, Boris S. Dolgovesov, Valerie V. Ovechkin, Sergei E. Chizhik, Nail R. Kaipov, Photorealistic imaging of digital terrains, freeforms and thematic textures in realtime visualization system Voxel Volumes, GraphiCon ’97, Moscow. - p.121–126.

2. Markus H. Gross, Roger Gatti, and Oliver G. Staadt. Fast multiresolution surface meshing. In Proc. of 14th International Conf. on Data Engineering, ICDE’98, IEEE, 1998. - p.550–557.

3. Gross M.H., Staadt O.G., Gatti R. Efficient Triangular Surface Approximations Using Wavelets and Quadtree Data Structures. IEEE Trans. on Visualization and Computer Graphics. Vol. 2, N 2, June 1996. - p.130-143.

4. Renato Pajarola. QuadTIN: Quadtree based Triangulated Irregular Networks. In Proceedings IEEE Visualization 2002, IEEE Computer Society Press, 2002. - p.395–402.

5. Вяткин С.И., Долговесов Б.С., Есин А.В. и др. Геометрическое моделирование и визуализация функционально-заданных объектов // Автометрия. 1999. N6. С.65-68.

6. Вяткин С.И. Моделирование и визуализация сложных поверхностей на основе скалярных функций возмущения // Сборник трудов ДонНТУ серии "Информатика, кибернетика и вычислительная техника". - 2009. - Вып. 10 (153). С. 105-111.

Поступила в редакцию 30.03.2010