РОССИЙСКАЯ АКАДЕМИЯ НАУК
СИБИРСКОЕ ОТДЕЛЕНИЕ
А В Т О М Е Т Р И Я
№ 1 2002

УДК 681.3.06

С. И. Вяткин, Б. С. Долговесов, Н. Р. Каипов
(Новосибирск)

ОТОБРАЖЕНИЕ ТЕКСТУРЫ НА ПЛОСКИЕ И КРИВОЛИНЕЙНЫЕ ПОВЕРХНОСТИ, СВОБОДНЫЕ ФОРМЫ И ОБЪЕМЫ

Рассматривается задача отображения двумерного массива текстуры на произвольно ориентированную плоскость в трехмерном пространстве и на криволинейные поверхности. Обсуждается способ отображения трехмерного массива текстуры на свободные формы и объемы.

Введение. Отображение текстуры на поверхности является эффективным методом повышения реализма в системах компьютерной графики. Применение текстур позволяет в первую очередь моделировать цветовой рисунок на поверхностях, а также прозрачность, резкие границы, движущиеся объекты и многие другие спецэффекты. Текстура значительно повышает визуальную сложность изображения при относительно малом количестве вычислений.

Термин «текстура» определяется в компьютерной графике как двумерное изображение, которое переносится на трехмерную плоскость или трехмерную криволинейную поверхность, или трехмерное изображение, отображаемое на трехмерное пространство.

Отображение текстуры включает в себя геометрическое отображение массива текстуры на поверхность и фильтрацию изображения, которая устраняет ступенчатые линии, муары и все, что в литературе по компьютерной графике определяется термином «элайсинг» (aliasing).

Процесс отображения текстурных карт на плоские грани включает два этапа. Первый этап – перспективное преобразование, т. е. вычисление координат текстурной карты (u, v), соответствующих координатам (X_s, Y_s) пиксела на экране. Плоскость параметризуется, если задать на ней единичные текстурные векторы **U** и **V** и точку привязки. Задача состоит в том, чтобы, зная координаты экрана (X_s, Y_s), получить текстурные координаты U и V соответствующей точки плоскости в системе координат наблюдателя. Это преобразование может быть записано в матричном виде [1]:

$$[UW, VW, W] = [X_s, Y_s, 1] \begin{vmatrix} A_u & A_\mathrm{v} & A_z \\ B_u & B_\mathrm{v} & B_z \\ C_u & C_\mathrm{v} & C_z \end{vmatrix}. \qquad (1)$$

Текстурные координаты связаны с экранными координатами дробно-линейной функцией

$$u = \frac{A_u X_s + B_u Y_s + C_u}{A_z X_s + B_z Y_s + C_z}, \tag{2а}$$

$$\mathrm{v} = \frac{A_\mathrm{v} X_s + B_\mathrm{v} Y_s + C_\mathrm{v}}{A_z X_s + B_z Y_s + C_z}, \tag{2б}$$

где (X_s, Y_s) – экранные координаты пиксела; A_i, B_i, C_i – элементы матрицы преобразования координат.

Второй этап отображения текстуры – фильтрация, необходимая для предотвращения элайсинга. Чаще всего применяются MIP-map (пирамидальные) текстурные карты, наиболее подходящие для аппаратной реализации [2].

Путем предварительной фильтрации получают набор квадратных текстурных карт с разным разрешением для каждого объекта. Каждой текстурной карте ставят в соответствие целочисленное значение уровня детальности (LOD – level of detail).

В зависимости от расстояния до грани и ее ориентации выбираются для работы две текстурные карты с соседними уровнями детальности. Критерием выбора является линейный размер проекции пиксела на грань. Если проекция пиксела накрывает менее двух текселов (texel – элемент текстурной карты), наблюдается элайсинг. У граней с углами наклона (между плоскостями грани и экрана), близкими прямому, текстурный рисунок сильно размывается. Этот эффект носит название *смаза* [1].

Затем в соответствии с текстурными координатами из каждой текстурной карты считываются по четыре тексела. Трилинейная интерполяция по этим восьми значениям завершает процесс фильтрации. Коэффициентами трилинейной интерполяции являются дробные части текстурных координат и уровня детальности. Путем видоизменения процесса фильтрации получают некоторые специальные эффекты. Например, можно получить текстурный рисунок с постоянной шириной контура.

1. **Вычисление уровня детальности текстуры.** Рассмотрим выбор уровня детализации текстурной карты. В общем случае проекция «круглого пиксела» на плоскость есть эллипс [1, 3], большую ось которого возьмем в качестве диаметра фильтра. Размер эллипса зависит от

– интервала выборки $\delta = \max(\delta x, \delta y)$;

– расстояния (D) от наблюдателя до точки пересечения луча зрения с плоскостью P;

– угла наклона плоскости к лучу зрения.

Большая ось эллипса лежит на линии пересечения двух плоскостей. Одна из них – текстурированная плоскость, а другая есть плоскость, в которой лежат нормаль к плоскости и луч зрения. Из этого следует, что $\varnothing = 2\delta D/\cos\theta$, где θ – угол между нормалью к плоскости и лучом зрения; $\cos\theta = d/D$ (d – расстояние от начала системы координат до плоскости, D – расстояние от наблюдателя до точки пересечения луча зрения с поверхностью объекта). Поэтому

$$\varnothing = 2\delta D^2/d \approx 2\delta Z^2/d. \tag{3}$$

2. **Трилинейная интерполяция.** Для получения окончательного результата, т. е. цвета и прозрачности в каждом пикселе, требуется произвести трилинейную интерполяцию [2]:

$$\text{result} = F\_\text{level} + \gamma(C\_\text{level} - F\_\text{level}), \tag{4а}$$

$$F\_\text{level} = F\_\text{bottom} + \beta_f(F\_\text{top} - F\_\text{bottom}), \tag{4б}$$

$$F\_\text{bottom} = f(f_u, f_v) + \alpha_f(f(f_{u+1}, f_v) - f(f_u, f_v)), \tag{4в}$$

$$F\_\text{top} = f(f_{u+1}, f_{v+1}) + \alpha_f(f(f_{u+1}, f_{v+1}) - f(f_u, f_{v+1})), \tag{4г}$$

где α_f, β_f – дробные части текстурных координат высокого уровня детальности; α_c, β_c – дробные части текстурных координат низкого уровня детальности; $c[c_u, c_v]$, $f[f_u, f_v]$ – значения цветовых компонент в точке $[u, v]$ для каждого из уровней детальности.

Исходные данные – восьмиразрядные значения компонент цвета и прозрачности. Коэффициенты билинейной интерполяции α и β для каждого из выбранных уровней детальности получаются из U и V адресов. Коэффициент линейной интерполяции между разными уровнями детальности γ выбирается в зависимости от наибольшего диаметра проекции пиксела (эллипса) на текстурную плоскость.

3. **Контурные текстуры.** Контурная текстура должна обеспечивать возможность создания и отображения контуров произвольной формы в рамках текстурной карты, не размывающихся при приближении. Область применения такой текстуры – отображение букв, символов, границ раздела и т. д. Перечислим основные свойства контурных текстурных карт:

– ширина контура составляет 1–2 пиксела;

– при приближении действительное значение LOD отрицательно;

– контур не размывается, ширина контура сохраняется;

– в области с неотрицательными значениями LOD контурные текстуры переходят в обычные, RGBT-текстуры;

– форма контура достаточно произвольна, т. е. для задания контуров требуется более одного бита на тексел.

Для обработки контурных текстурных карт необходим канал прозрачности. Это позволяет получать полноцветные изображения внутри контуров. Однако для получения полноценной картины потребуется двойной слой текстурированных граней: один для изображения внутри контура, другой – вне его.

После билинейной интерполяции, такой же, как и в остальных каналах, для выделения контура производятся следующие вычисления:

$$T = \frac{t - 0{,}5}{P_s} + 0{,}5, \tag{5}$$

далее следует коррекция:

$$T=\begin{cases}0, & \text{если } T<0;\\ T, & \text{если } 0<T<1;\\ 1, & \text{если } T>1,\end{cases} \tag{6}$$

где T – окончательный результат; t – результат билинейной интерполяции; P_s – размер проекции пиксела.

Дополнительно производится операция выделения контура:

$$T=(t-0{,}5)\ll \log_2 P_s+\gamma'(t-0{,}5)\ll \log_2 P_s+0{,}5, \tag{7}$$

где γ' – дробная часть размера проекции пиксела.

4. **Текстурные адреса.** Для вычисления текстурного адреса необходимы текстурные координаты (2а), (2б), уровень детальности пиксела, величина общего порядка текстурных координат и смещение локальных и глобальных текстурных координат относительно друг друга. Далее каждая из текстурных координат разделяется на четыре числа.

Коэффициенты билинейной интерполяции α и β используются для трилинейной интерполяции текстуры:

$$f_u=((f_{iu}-d_u)\,\&\,0111)+(d_u\,\&\,011), \tag{8а}$$

$$c_u=((c_{iu}-d_u/2)\,\&\,011)+((d_u/2)\,\&\,011), \tag{8б}$$

где f_{iu}, f_{iv} и c_{iu}, c_{iv} – текстурные координаты текущего пиксела, масштабированные в соответствии с уровнем детальности; d_u (d_v) – смещение сетки локальных текстурных координат относительно глобальных (имеют смысл только для сжатых текстурных карт и равны нулю для несжатой текстуры). Для f_v, c_v вычисления аналогичны. Текстурный адрес формируется из величин f_u, f_v, c_u, c_v.

5. **Отображение текстуры на свободные формы и объемы.** Дадим определение текстуры в «объемном» смысле, исходя из ее первоначального определения в «поверхностном» варианте (описано выше). Кроме того, однозначно определим основные понятия.

Текстура – это детализация поверхности или объема отображаемого объекта. Здесь существенно то, что не оговаривается ее конкретное строение, т. е. текстура в общем случае не обязательно двумерный массив текселей.

Отображаемый объект – это вся трехмерная сцена или ее элемент, несущий информацию о своей форме [4]. Например, сцена, представленная двумя эллипсоидами, с одной стороны, может рассматриваться как изображение двух независимых объектов, с другой стороны, такие эллипсоиды можно рассматривать как объект-объединение двух эллипсоидов; эти две точки зрения альтернативны. Тогда в терминах объектов текстура – это элемент сцены, изменяющий свойства другого объекта и сам не отображающийся при растрировании. Например, раскрашенный шар будет представлен двумя объектами: формой в виде эллипсоида и объектом-текстурой, содержащим ссылку на массив цветов, каким-то определенным образом отображаемый на эллипсоид с использованием линейной, цилиндрической, сферической или какой-нибудь другой параметризации.

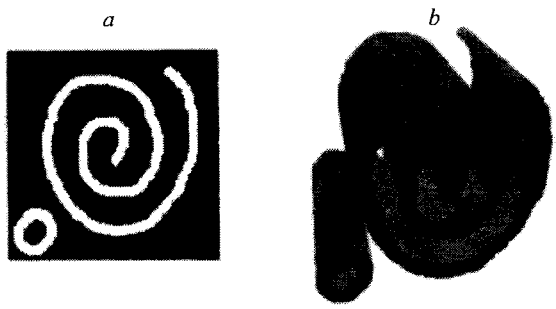

Рис. 1. Двумерный массив (*a*), инкапсулированный в объект-массив (*b*)

Свойство отображаемого объекта есть любой параметр, назначенный этому объекту, который используется при вычислении результирующего цвета пиксела на экране: цвет объекта, весовые коэффициенты диффузной и отражающей составляющих модели освещения Фонга, полупрозрачность и т. д. Текстура, таким образом, может влиять на любой параметр отображаемого объекта.

Кроме объектов, заданных аналитически, например квадрик, существует возможность отображать объекты, представляющие собой скалярные массивы данных [5]. Это значительно расширяет возможности и области применения системы в следующих случаях.

1. Визуализация трехмерных массивов, полученных в результате сканирования реальных объектов в реальном пространстве (томография и т. д.).

2. Визуализация научных данных, полученных в результате расчетов или измерений.

3. Визуализация метеорологических данных (распределение температуры, влажности и т. д.).

4. Визуализация двумерной сетки высот без предварительной триангуляции.

Скалярные массивы хотя и требуют значительных затрат памяти (порядка N^2 или N^3 для двумерных и трехмерных массивов соответственно), но по сравнению с эквивалентным их представлением в виде множества треугольников они все же более компактны и удобны в использовании и манипулировании.

Пусть имеется объект, представляющий собой какой-то скалярный массив в каком-то формате, известном только самому объекту. Определим этот объект как *объект-массив*. В итоге в системе основные примитивы будут представлены двумя классами объектов. Один класс – это аналитически описанные примитивы (квадрики), другой – массивы скалярных данных. Оба класса имеют более компактное описание по сравнению с полигональным заданием объектов (треугольниками).

Объект-массив может быть реализован несколькими способами. Он может заключать в себе одномерный, двумерный (рис. 1) или трехмерный массив (рис. 2), а также может являться

Рис. 2. Визуализация объекта-массива, инкапсулирующего в себе трехмерный массив данных

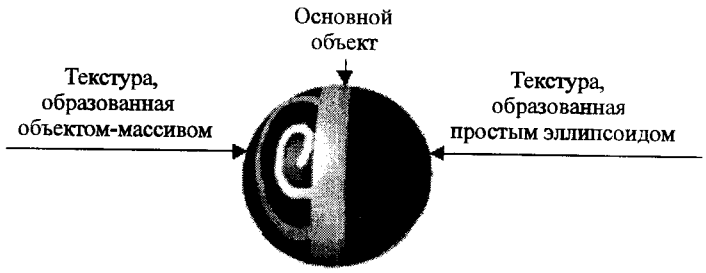

Рис. 3. Текстура, образованная объектом-массивом и эллипсоидом

сеткой высот [4]. (На рис. 1 и 2 представлены нефильтрованные изображения.)

По данному в этой работе определению текстура – это объект (не отображаемый), изменяющий свойства другого объекта. Особенностью текстуры является то, что ею может быть любой объект. Для этого было предложено незначительное изменение схемы обработки данных и построения дерева сцены, которое не затрагивало самого алгоритма растрирования [6, 7]. На самом последнем уровне рекурсии деления вычисляется ближайший видимый воксел и запрашиваются у отображаемого объекта (сцены) параметры этого элемента пространства для их использования при вычислении цвета пиксела. Дополнением к структуре данных при учете текстуры является то, что каждый объект может содержать ссылку на другой объект, являющийся для него текстурой (рис. 3). Если этой ссылки нет, то объект на запрос о значении своих свойств выдает значение, определенное для этого объекта по умолчанию. Иначе, параметры объекта в текущем вокселе подвергаются изменениям со стороны объекта-текстуры, на которую ссылается наш объект.

Важным при наложении текстуры на объект является преобразование координат из текстурного пространства (U, V, W) в пространство объекта (X, Y, Z). В нашем случае параметризация нужна только для объектов-массивов. Чтобы иметь возможность накладывать текстуру на поверхность и объем отображаемого объекта, объекты-массивы дополнены функцией преобразования координат из модельного пространства M в текстурное пространство T. При запросе у объекта-массива значения текстуры происходит пре-

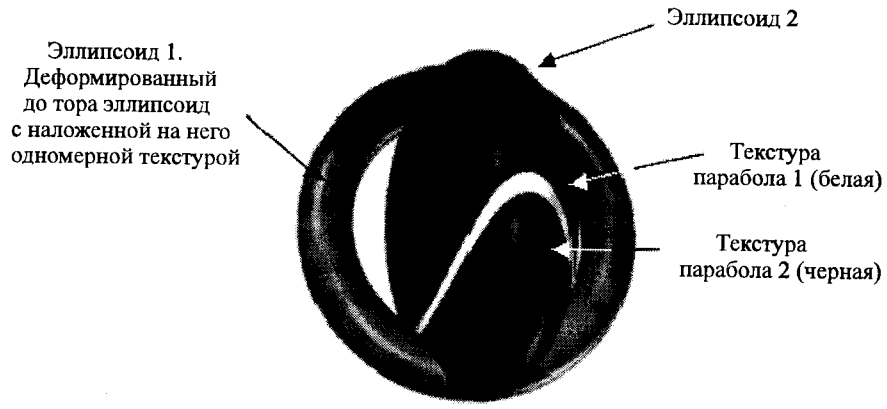

Рис. 4. Пример сцены с двумя текстурированными объектами

Рис. 5. Логическая схема сцены

образование координаты текущего воксела из M в T: $(X, Y, Z) \rightarrow (U, V, W)$, затем эти координаты используются как адрес в массиве. Если, например, массив двумерный, то одна координата не используется. Реализованы три вида преобразований объектных координат в текстурные.

1. *Линейное*:

$$U = (1+X)/2, \quad V = (1+Y)/2, \quad W = (1+Z)/2.$$

(Преобразование из двоичного куба в единичный).

2. *Цилиндрическое*:

$$U = \operatorname{arctg}(Y/X)/2\pi, \quad V = (1+Y)/2, \quad W = \sqrt[2]{(X^2+Y^2)}.$$

3. *Сферическое*:

$$U = \operatorname{arctg}(Y/X)/2\pi, \quad V = \operatorname{arctg}(\sqrt[2]{(X^2+Y^2)}/Z)/2\pi, \quad W = \sqrt[2]{(X^2+Y^2+Z^2)}.$$

Приведем пример структуры данных для конкретной сцены. На рис. 4 показано ее изображение, а на рис. 5 изображена логическая схема сцены.

Заключение. Основные производители графических акселераторов стремятся повышать реалистичность уже не только за счет количества полигонов. Такие технологии как Bump Mapping и Environment Mapped Bump Mapping позволяют на порядок увеличить реалистичность изображения, не требуя для этого повышения количества геометрических примитивов.

В настоящий момент все эти эффекты моделируются за счет специфических приемов. Один из приемов – бугорковое текстурирование (Bump Mapping), описанное в [8], недостатками которого являются сильные ограничения на условия его применения. Например, в случае бугоркового текстурирования реализм достигается только при углах зрения, близких к прямому. Совсем недавно появилась новая технология – рельефное текстурирование (Relief Texture Mapping) [9], которое устраняет ограничение на угол зрения. Данная технология является трехмерной в отличие от бугоркового текстурирования, создающего лишь иллюзию трехмерных деталей поверхности.

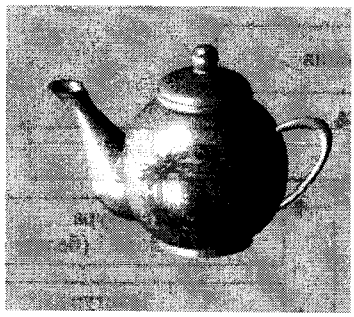

Рис. 6. Текстурированный функционально заданный объект свободной формы

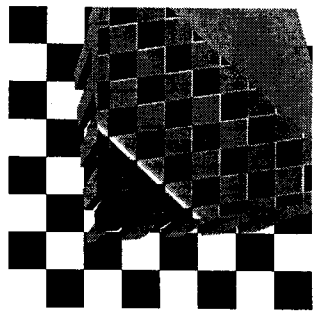

Рис. 7. Примеры двумерной и трехмерной текстур

Способы отображения текстуры, изложенные в работе, реализованы программно. Рис. 1–4, 6 и 7 иллюстрируют полученные результаты. Предложенные способы отображения текстур пригодны для аппаратной реализации. Резюмировать вышеизложенное можно следующим образом. Повышение реализма идет уже не только за счет количества примитивов, которыми представлена сцена, а за счет моделирования специальных эффектов окружающей среды и мелких деталей поверхностей объектов сцены.

СПИСОК ЛИТЕРАТУРЫ

1. **Heckbert P. S.** Survey of texture mapping // IEEE Comput. Graph. and Applicat. 1986. **6**, N 11. P. 56.
2. **Williams L.** Pyramidal parametrics // Comput. Graph. 1983. **12**, N 4. P. 270.
3. **Ковалев А. М., Тарасов Ю. В.** Текстура на произвольно ориентированных плоских поверхностях // Автометрия. 1988. № 6. С. 9.
4. **Vyatkin S. I., Dolgovesov B. S., Ovechkin V. V. et al.** Photorealistic imaging of digital terrains, freeforms and thematic textures in realtime visualization system Voxel-Volumes. GraphiCon'97. М.: МГУ, 1997. С. 35.
5. **Vyatkin S. I., Dolgovesov B. S., Yesin A. V. et al.** Voxel-Volumes volume-oriented visualization system: Intern. Conf. on Shape Modeling and Applications (March 1–4, 1999, Aizu-Wakamatsu, Japan) // IEEE Comput. Soc. Los Alamitos, California, 1999. P. 234.
6. **Vyatkin S. I., Dolgovesov B. S., Chizhik S. E.** Synthesis of virtual environment with object-space recursive subdivision. Graphicon'98. Proc. /Eds. S. Klimenko et al. М.: МГУ, 1998. С. 54.
7. **Vyatkin S. I., Dolgovesov B. S., Guimaoutdinov O. Y.** Synthesis of virtual environment using perturbation functions: World Multiconference on Systemics, Cybernetics and Informatics Proceedings (Orlando, Florida, USA, July 22–25, 2001) // Emergent Computing and Virtual Engineering. 2001. V. III. P. 350.
8. **Blinn J. F.** Simulation of wrinkled surfaces // Comput. Graph. 1978. **12**, N 3. P. 263.
9. **Oliveira M. M., Bishop G., McAllister D.** Relief Texture Mapping // Proc. SIGGRAPH 2000. New Orleans, Louisiana, July 23–28, 2000. P. 324.

Институт автоматики и электрометрии СО РАН,
E-mail: sivser@mail.ru

Поступила в редакцию
26 апреля 2001 г.