Интерактивный выбор объектов трёхмерной сцены

Е.И. Коростелев, М.А. Городилов, Б.С. Долговесов
Институт Автоматики и Электрометрии СО РАН, Новосибирск, Россия
kore3d@gmail.com

Аннотация

В статье предлагается метод выбора трёхмерных объектов виртуальной сцены для систем визуализации реального времени. Метод использует один бит буфера трафарета и позволяет определять порядковый номер объёкта, которому принадлежит некоторый пиксель итогового изображения. Таким образом, он позволяет определять объекты под курсором мыши, что позволяет интерактивно взаимодействовать с ними. Для этого не требуется дополнительных проходов визуализации, и отсутствуют ограничения на используемые при визуализации материалы (процедурные сетки, полупрозрачность, альфа-тест).

Ключевые слова: *выбор объектов, буфер трафарета, визуализация реального времени.*

1. ВВЕДЕНИЕ

Большинство систем виртуальной реальности предоставляют возможность интерактивного взаимодействия с растеризуемыми виртуальными объектами. Как правило, взаимодействие осуществляется путём выбора конкретного объекта курсором мыши, а выбранный объект выделяется специальным образом. Кроме этого, в презентационных и обучающих системах [1] существует необходимость рисовать поверх объектов, для чего используется аналогичный механизм выбора.

В системах визуализации, где число доступных для выбора объектов незначительно, как правило, применяется метод выбора объектов путём нахождения ближайшего пересечения луча с набором доступных объектов[2]. Этот алгоритм реализуется на центральном процессоре и не позволяет точно выделять динамически изменяющиеся объекты (например, процедурно-генерируемые сетки).

Одним из методов определения порядкового номера (порядок визуализации) объекта под курсором является: отдельный проход растеризации всех видимых объектов сцены с упрощенными материалами и запись в буфер цвета (как правило, одноканальный 32-х битный) идентификатора объекта [3]. Этот метод использует графический процессор, но имеет ряд недостатков. Во-первых, он требует повторную растеризацию всех объектов (списка батчей), что накладно в случае динамически изменяющихся сеток (например, скелетная анимация). Во-вторых, в «простом» материале, используемой на отдельной стадии, необходимо учитывать свойства материалов, которые отбраковывают часть поверхности (например, инструкция discard в шейдере) или дают альфа-канал (альфа-тест).

Авторами статьи предлагается метод, лишенный изложенных выше недостатков и обладающий определенными преимуществами: простота интеграции в существующую систему, производительность, возможность реализации для видеокарт уровня DirectX 9. Несущественными недостатками предлагаемого метода является необходимость использовать 1 бит буфера трафарета [4] (stencil buffer), теста трафарета (stencil test) и дополнительной видеопамяти (до 32 МБ на разрешение сверхвысокой чёткости 4K UHDTV).

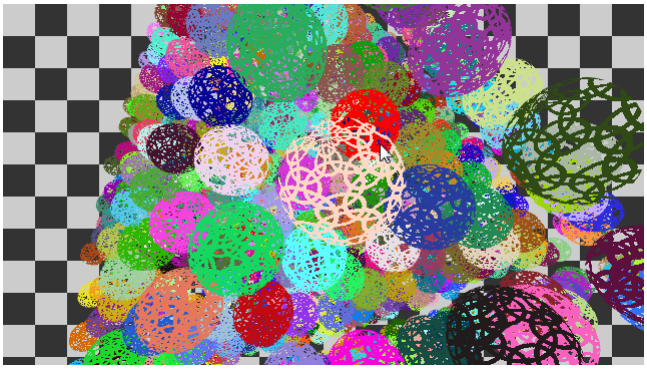

Рис 1: Выбор одного из 1000 виртуальных объектов

Пример использования разработанного метода приводится на рисунке 1. На изображении представлен результат визуализации тысячи трехмерных объектов (разноцветных сфер), поверхность которых определяется специальной текстурой (отбраковка части поверхности), а цвет задан случайно. Объект, находящийся под курсором, выделен специальным цветом.

2. АЛГОРИТМ

Основой метода является использование буфера и теста трафарета. Буфер трафарета используется для перезаписи 1 бита после визуализации каждого объекта (вызов отрисовки). Поскольку требуется перезаписывать только 1 бит, данный метод может быть скомбинирован с другими алгоритмами, использующими трафарет, либо применяться без ограничений. Большинство алгоритмов визуализации не используют тест трафарета и ограничиваются использованием буфера и теста глубины (буфера формата D24SX, где D24 – 24 бита для записи глубины). Таким образом, достаточно использовать буфер формата D24S8.

Дополнительно, для хранения идентификаторов растеризуемых объектов, потребуется один буфер, соответствующий размеру кадра (обозначим его *PickBuffer*). Для него достаточно формата R16G16. В красный канал (R16) будет записываться идентификатор объекта.

При использовании предлагаемого метода визуализация всех объектов производится как обычно, без ограничений. Алгоритм поддерживает буферы с несколькими «семплами» и не влияет на поддержку алгоритмов устранения ступенчатости.

Рассмотрим общую схему работы алгоритма:

1. Начало кадра: инициализируем буфер трафарета значением 0.
2. Визуализация объектов сцены: включаем запись в буфер трафарета и пишем 1 бит, если пиксель прошёл тест глубины (либо тест выключен)

2.1 После каждого вызова отрисовки временно переключаем буфер кадра на *PickBuffer* и с включенным тестом трафарета на проверку бита с шага 1 рисуем прямоугольник в 1 пиксель (размер области, в которой необходимо определить ближайший объект). При этом, во время проверки сбрасываем этот бит.

3. Конец кадра: пересылаем содержимое *PickBuffer* в системную память.
4. Декодирование области *PickBuffer* (распаковка идентификатора) – получение номера объекта

На шаге 2.1 может быть выбрана область произвольного размера, но её существенное увеличение приведет к падению производительности, поскольку потребуется обновлять больше значений в видеопамяти. Каждая видеокарта имеет ограничение скорости заполнения пикселами (филлрейт).

Содержимое буфера идентификаторов (*PickBuffer*) для полного кадра тестовой сцены представлено на рисунке 2. На изображении для наглядности заполнены все пиксели, но, в случае использования однопиксельной области, в буфере обновляется только 1 пиксель. Кроме этого, для лучшего визуального различия содержимого буфера цвета были переназначены и использован красный цвет для подсветки текущего выбранного объекта.

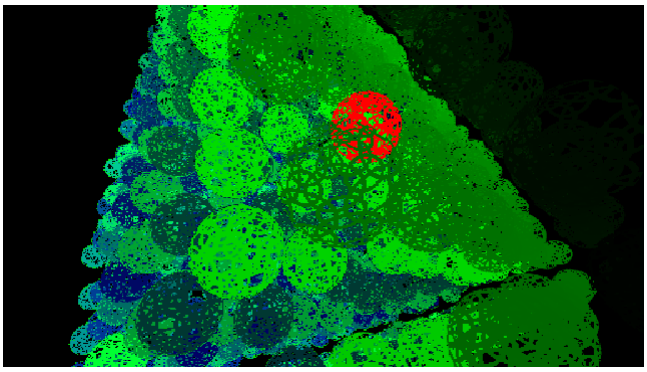

Рис 2: Буфер идентификаторов 3d-объектов полного кадра

При использовании мультисемплирования буфер идентификаторов содержит несколько субпикселей на пиксель. Перед передачей его в системную память производится преобразование субпикселей в пиксель. Данный механизм поддерживается аппаратно, но плохо конфигурируется и, при настройках по умолчанию, даёт усредненные значения по границам объектов, что нельзя интерпретировать как идентификаторы. На большинстве видеокарт 8-летней давности поддерживается возможность задать маску семплирования, что позволяет исключить преобразование значений. В случае отсутствия возможности маскировать какой-то один субпиксель, допускается использование области из 9 пикселей на шаге 2 и отбраковке значений, если в окрестности не найдено 2-3 совпадения.

Копирование *PickBuffer* в системную память (третий шаг алгоритма) следует выполнять с задержкой, как минимум, в 1 кадр, т.е. на текущем кадре считывать предыдущий. При несоблюдении этого условия возникает точка синхронизации между графическим и центральным процессором, что отрицательно сказывается на производительности. Причина в том, что центральный и графический процессор работают асинхронно и, к моменту выполнения шага 3 центральным процессором, данные будут еще не готовы. Кроме этого, все команды графическому процессору буферизируются и могут задерживаться на несколько кадров.

Небольшое запаздывание выбора объекта под курсором мыши (из-за задержки считывания *PickBuffer*) устраняется путём прогнозирования перемещения курсора. Для этого используются значения ускорения и скорости движения мыши за несколько предыдущих кадров, а для текущего кадра берется прогнозируемая позиция.

3. РЕЗУЛЬТАТЫ

Реализация алгоритма тестировалась на различном аппаратном обеспечении. При этом типовым случаем считалось наличие большого набора объектов и использование алгоритмов устранения ступенчатости (мультисемплирование). Приводимые далее результаты были получены в реализации алгоритма на Direct3D11 и исполнение его на видеокарте с аппаратной поддержкой DirectX10 (функциональный профиль 10.0 устройства Direct3D11). Кроме этого, работоспособность алгоритма проверена в рамках функций Direct3D9 (и функционального профиля 9.3 устройства Direct3D11). Результаты замера производительности для реализации Direct3D11 приведены в таблице 1.

Видеокарта \ N	512	1000	2197
NVIDIA 8800 GTX	3 мс	4 мс	5 мс
AMD FirePro W8000	5 мс	13 мс	26 мс

Табл. 1: Производительность при выборе среди N объектов

Как видно, из таблицы наиболее эффективно метод работает на видеокартах NVIDIA. На результат тестирования при этом влияет, как версия драйвера, так и поколение видеочипа. Поскольку в большинстве случаев не требуется выбирать объект из десятков сотен других, производительности достаточно для решения большинства задач, где необходимо интерактивное взаимодействие с трёхмерными объектами.

Разработанный метод также позволяет выбирать объекты последовательно, т.е. выбирать объекты не прошедшие растеризацию или последовательно прятать ближайшие. Такой механизм удобен при необходимости выбирать объекты, находящиеся позади полупрозрачных. Данная функция реализуется посредством составления «чёрного» списка объектов, для которых игнорируется запись в буфер идентификаторов.

Возможность рисования поверх поверхности объекта после выбора некоторого объекта обеспечивается путём дополнительной растеризации этого объекта в конце кадра. Это позволяет определить текстурные координаты выбранной поверхности и реализуется с использованием уже заполненного на этапе подготовки кадра буфера глубины.

4. ЗАКЛЮЧЕНИЕ

В статье предложен эффективный алгоритм выбора объектов, использующий аппаратные возможности современных и старых поколений графических процессоров. Разработанная технология применяется в интерактивной системе визуализации реального времени, что позволяет предоставить пользователю возможность выбирать объекты, менять их свойства, а также рисовать на поверхности виртуальных объектов.

5. БИБЛИОГРАФИЯ

[1] Долговесов Б.С. Системы виртуальной реальности для космических тренажерных комплексов и интерактивных презентаций // Сборник материалов конференции «Пилотируемые полеты в космос», ИМБП РАН, 2011. С. 66.

[2] Brian Hook, Mouse Ray Picking Explained, http://bookofhook.com

[3] D. Cantor et el., WebGL Beginner's Guide, 2012, pp.257-285

[4] J. Zink et el. Practical Rendering and Computation With Direct3D 11, 2011, pp.241-258

Interactive object picking by stencil buffer technique

Abstract

A new approach for fast object picking is presented in the article. Described technique uses a stencil test. No additional rendering passes are needed. So this technique allow to select any meshes (procedural, alpha blended, etc) and works fine with multisample render targets. And also it's implementable for d3d9-capable hardware.

Keywords: *object picking, stencil buffer, real-time rendering.*

About the author

Evgeny I. Korostelev is a Ph.D. student at Institute of Automation and Electrometry SB RAS. His contact email is kore3d@gmail.com

Mikhail A. Gorodilov is a Ph.D. student at Institute of Automation and Electrometry SB RAS. His contact email is gorodilovm@gmail.com

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