УДК 004.925

Погорелов Д.А., студент 1 курса магистратуры, факультет «Программное обеспечение ЭВМ и информационные технологии», МГТУ им. Н.Э. Баумана, Россия, г. Москва

Смелкова Е.А., старший преподаватель кафедры «Английский язык для приборостроительных специальностей», МГТУ им. Н.Э. Баумана, Россия, г. Москва

Таразанов А.М., Меркулов Д.В., студенты 1 курса магистратуры, факультет «Программное обеспечение ЭВМ и информационные технологии», МГТУ им. Н.Э. Баумана, Россия, г. Москва

Иксарица Н.И., студент 2 курса, факультет «Программное обеспечение ЭВМ и информационные технологии», МГТУ им. Н.Э. Баумана, Россия, г. Москва

БУЛЕВЫ ОПЕРАЦИИ НА ТРЁХМЕРНЫХ МОДЕЛЯХ В КОМПЬЮТЕРНОЙ ГРАФИКЕ

Аннотация: рассмотрены основы Constructive Solid Geometry (CSG), алгоритмы, позволяющие реализовать CSG-модели, подробно описана возможная реализация алгоритма, основанного на работе с полигональными моделями, предложены методы оптимизации приведённой реализации для уменьшения потребления памяти и процессорного времени, позволяющих производить построение CSG-моделей в реальном времени.

Ключевые слова: компьютерная графика, пересечение многогранников, трёхмерная геометрия, ограничивающие объёмы, Constructive Solid Geometry, триангуляция, булевы операции.

Annotation: *The fundamentals of Constructive Solid Geometry (CSG) are described in this paper. These are algorithms for implementing CSG models. The possible implementation of an algorithm based on working with polygonal models is described in detail. Methods of optimization of the provided implementation are proposed. They can be used to reduce the consumption of memory and process time allowing to build CSG models in CPU time.*

Key words: *computer graphics, polygon intersections, three-dimensional geometry, bounding volumes, Constructive Solid Geometry, triangulation, Boolean operations.*

Введение. CSG (Constructive Solid Geometry) – техника, активно применяемая в компьютерной графике, существенно облегчающая восприятие сложных геометрических моделей. При помощи CSG можно представить модель любой формы, используя булевы операции над выпуклыми примитивами [1].

Данная тематика является важной для людей, работающих с компьютерной графикой, в частности, в области трёхмерного моделирования.

В первой части настоящей работы рассмотрены возможные алгоритмы построения CSG-моделей, во второй подробнее рассмотрен алгоритм работы с полигональными моделями, в третьей даны соответствующие выводы.

Алгоритмы построения CSG-моделей. Существуют следующие алгоритмы построения CSG-моделей: работающие в пространстве изображения и работающие в пространстве объекта. Вторые являются более гибкими, так как обрабатывают модель до её вывода на экран. К ним принадлежат алгоритмы, работающие с полигональными моделям, представляющие для нас интерес, из-за своей гибкости.

Алгоритм, работающий с полигональными объектами. Главным условием данного алгоритма является полигональное представление объектов, поэтому он может работать с невыпуклыми примитивами. Также он позволяет полностью абстрагироваться от того, выполнение какой булевой операции нам требуется, и работать непосредственно с объектами. Для каждого введённого примитива (одному из двух) определяется два набора треугольников, которые пространственно-ориентированы относительного второго примитива, поэтому для реализации булевых операций достаточно выбрать по одному набору для каждого примитива, и, отрисовав их, получить результат.

Алгоритм можно разбить на три основных этапа.

Этап 1. Нахождение линии пересечения двух заданных полигональных примитивов, для чего необходимо найти все отрезки попарных пересечений, входящих в них треугольников. Данная задача сводится к нахождению отрезка пересечения двух треугольников в пространстве, которая может быть решена методом [2]. Предварительно проверив граничные объёмы каждого из примитивов на взаимопересечение, можно уменьшить количество проверяемых пар полигонов, то есть, если хотя бы в одном измерении минимальное значение координаты одного треугольника больше максимального, то такие треугольники точно не пересекаются (рис. 1).

Далее необходимо провести проверку положения каждого треугольника относительно плоскости другого, подставив в уравнение плоскости проверяемого треугольника координаты вершин другого: $f(p) = (N, q_0) - (N, p)$, где N – нормаль текущего треугольника, q_0 – любая точка, принадлежащая этому треугольнику, p – точка второго треугольника. С помощью $f(p)$ можно определить отклонение точки от заданной плоскости, если значения для всех вершин другого треугольника одного знака, то проверяемый треугольник лежит по одну сторону от плоскости другого, иначе, пересекает.

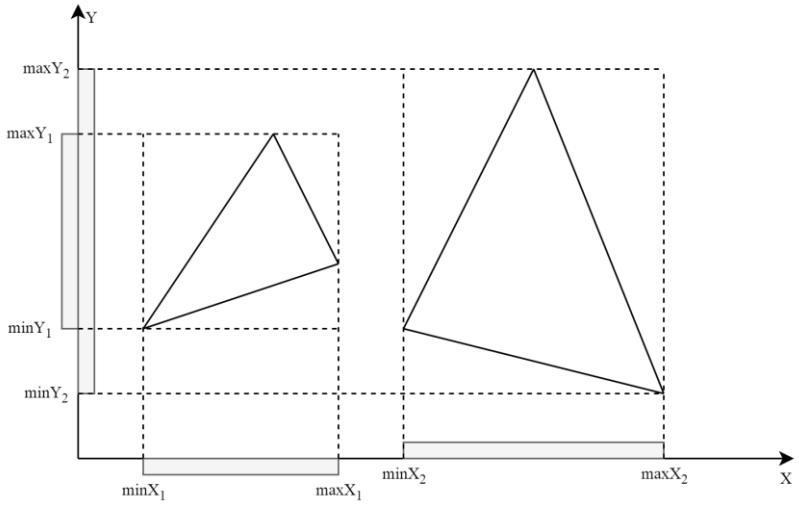

Рисунок 1 – Сравнение граничных объёмов двумерных треугольников.

Этап 2. После нахождения линии пересечения двух 3D-примитивов необходимо разбить каждый треугольник, которыми образован данный примитив, на группы треугольников таким образом, чтобы каждый из отрезков являлся стороной двух смежных треугольников (рис. 2).

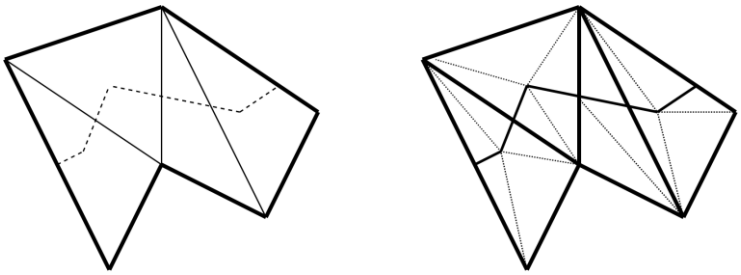

Рисунок 2 – Разбиение треугольников вдоль линии пересечения.

Для непосредственного разбиения треугольников на группы треугольников (триангуляции) необходимо воспользоваться итерационным методом триангуляции Б.Н. Делоне [3]. Так как линия пересечения двух треугольников представляет из себя две точки, лежащие в каждом из треугольников, то триангуляцию одного такого треугольника можно представить, как рисунке 3.

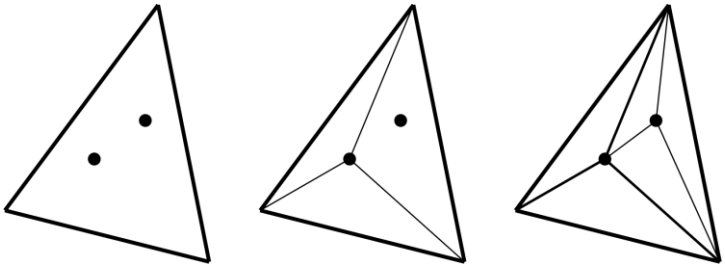

Рисунок 3 – Итеративная триангуляции.

Результатом разбиения всех треугольников являются две группы треугольников для каждого примитива. Однако, может возникнуть ситуация, когда один треугольник пересекают два и более других треугольников, тогда при разбиении две точки другого треугольника могут оказать в двух разных примитивах. Данная проблема решается поиском пересечения линии, соединяющей точки пересечения и одной из стороны рассматриваемого примитива, после чего задача сводится обратно к итеративной триангуляции, рассмотренной выше.

Этап 3. Последним этапом является решение задачи пространственной локализации треугольника одного примитива, относительно другого. Так как после выполнения предыдущего этапа все пересекающиеся треугольники были разделены по линиям пересечения, для определения положения треугольника достаточно локализовать его барицентр, для чего необходимо воспользоваться методом бросания лучей (ray-casting) [4]. За точку испускания луча принимаем центр масс треугольника, а за направление – нормаль к его плоскости.

Для подсчёта количества пересечений необходимо проверить пересечение луча с каждым треугольником другого примитива и подсчитать количество пересечений с треугольниками, расположенными внутри. Также необходимо не учитывать пересечение со сторонами треугольников, так как может возникнуть ложная ситуация, когда один луч дважды пересекает одно и тоже ребро в разных треугольниках.

Для обнаружения пересечений лучей с треугольниками используется алгоритм [5], так как он работает с барицентрическими координатами.

Стоит отметить, что ray-casting может быть ускорен при помощи BVH-деревьев, так как тогда пересечение будет проверяться не для всех треугольников, а только попавших в определённый граничный объём.

Результатом данного этапа и всего алгоритма является связь ориентации треугольников и булевых операций. Относительно того, как расположен тот или иной треугольник (внутри/снаружи) и какая булева операция требуется, те или иные треугольники (таблица 1) будут отображены или же скрыты при визуализации.

Таблица 1.

Связь ориентации треугольников с булевыми операциями

	Примитив A	Примитив B
$A \cup B$	снаружи	снаружи
$A \cap B$	внутри	внутри
$A \setminus B$	снаружи	внутри
$B \setminus A$	внутри	снаружи

Заключение. В данной статье были рассмотрены основы CSG, рассмотрены возможные алгоритмы построения CSG-моделей, также описан подробно вариант реализации алгоритма, работающего с полигональными моделями. Предложены улучшения алгоритмов, реализующих данную задачу.

Использованные источники

1. R. Banerjee and J. Rossignac, Topologically exact evaluation of polyhedra defined in CSG with loose primitives, to appear in Computers Graphics Forum, Vol. 15, No. 4, pp. 205-217, 1996
2. Müller T. A fast triangle-triangle intersection test // Journal of Graphics Tools. 1997. V. 2. I. 2. P. 25-30.
3. Скворцов А.В. Триангуляция Делоне и ее применение // Томск:ТГУ. – 2002. – 128 с.
4. ZAJÍČEK, Petr Acceleration of Ray-Casting for CSG scenes. Master thesis, MFF UK, 2012.
5. Mцller T., Trumbore B. Fast, minimum storage ray-triangle intersection // Journal of Graphics Tools, 1997. V. 2. I. 1. P. 21-28.