

Линия пересечения двух объектов

Жумабаев С.К., Шикуть А.В., МГТУ им. Н.Э. Баумана
zhumabaevsk@student.bmstu.ru, alla.shikut@rambler.ru

Аннотация

Настоящая статья посвящена методам оптимизации процесса определения линии пересечения двух объектов. Приведен общий алгоритм поиска линии, по которой пересекаются две полигональные модели объектов. Проанализированы основные этапы алгоритма и предложены способы, позволяющие ускорить входящие в них вычисления.

1 Введение

Поиск пересечений трехмерных объектов играет значительную роль в нескольких обширных областях таких как пространственное ориентирование [Egenhofer, Franzosa, 1991], рендеринг в реальном времени [Tropp, Tal, Shimshoni, 2006], геология [Caumon, Sausse, 2009], геометрическое моделирование [Houghton, Emmett, Factor, 1985] и компьютерное зрение. Обнаружение возможного пересечения между объектами может быть основано на триангулированных границах объектов, что приводит к вычислению пересечения треугольник-треугольник.

В данной статье будет приведено полное описание алгоритма по нахождению линии пересечения между двумя пространственными объектами и описан один из способов оптимизации указанного процесса. Эта работа непосредственно применима к пространственному мышлению и геометрическому моделированию CAD/CAM, где оптимизация алгоритмов необходима для станков с ЧПУ.

2 Принципиальная схема алгоритма

Для выполнения основной задачи, освещаемой в данной работе, а именно - поиска линии пересечения двух объектов, необходимо выполнить определенную последовательность действий, включающую в себя два главных пункта и несколько подпунктов.

2.1 Определение факта пересечения

Первым действием необходимо установить существование факта пересечения двух рассматриваемых объектов. Для этого достаточно заключить каждую из моделей в минимальную

выпуклую объемлющую оболочку, свести к двумерному пространству и проверить наложение описывающих прямоугольников друг на друга. Данный вопрос подробно рассмотрен в работах [Вайнштейн, 1990, Скворцов, Мирза, 2006] и сводится к тривиальному решению.

2.2 Формирование линии пересечения

Для успешной реализации второй, и основной части алгоритма необходимо выделить две подзадачи, успешная работа каждой из которых необходима для правильного нахождения линии пересечения двух рассматриваемых объектов.

В рамках данной работы мы рассматриваем объекты, представленные в виде наборов полигонов, каждый из которых, в свою очередь, является треугольником. Таким образом для получения искомой линии пересечения двух объектов в целом необходимо и достаточно найти линии, по которым пересекаются полигоны первого объекта с полигонами второго.

Описанные линии, получающиеся при пересечении полигонов двух объектов, представляют собой множество отрезков. Стоит отметить, что в образовании подобного множества могут участвовать как все полигоны обоих или одного объектов, так и только часть из всей мириады полигонов (см. Рис. 1).

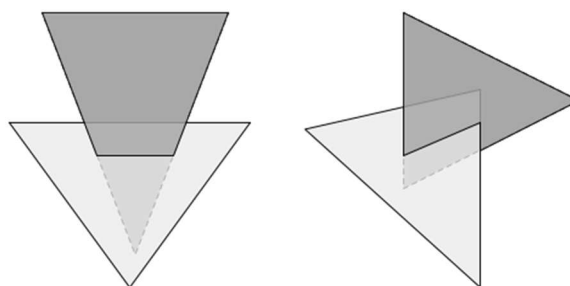


Рис. 1. Варианты пересечения треугольников

Как уже упоминало ранее, мы представляем каждый полигон каждого из рассматриваемых объектов в виде двумерного треугольника, который, в свою очередь, является плоскостью по определению. Исходя из этого логично применить типичный алгоритм поиска пересечения линии и плоскости.

Линия L и плоскость P могут находиться в двух возможных состояниях в пространстве относительно друг друга: либо они пересекаются в единственной точке, либо они параллельны. Рассмотрим указанные два варианта состояния линии L и плоскости P по очереди.

Первым действием необходимо определить линию L в пространстве: пусть она задана параметрическим уравнением (1):

$$P(S) = P_0 + S(P_1 - P_0) = P_0 + SU \quad (1)$$

Также нужно задать плоскость P . Пусть она определяется принадлежащей ей точкой V_0 и вектором нормали $n = (a, b, c)$ к ней.

Итак, проверим являются ли линия L и плоскость P взаимно параллельными. Для этого необходимо лишь вычислить векторное произведение векторов $\vec{n}$ и $\vec{u}$. Если результат будет равен нулю, то плоскость P и линия L либо являются параллельными, либо никогда не пересекаются, либо линия L полностью лежит в плоскости P .

Для исключения третьего варианта необходимо провести окончательную проверку на определение факта полного заключения линии L в плоскость P . Для этого воспользуемся условием (2):

$$n(P_0 - V_0) = 0 \quad (2)$$

Если при проверке параллельности линии и плоскости мы получили результат отличный от нуля, то это означает, что линия L и плоскость P имеют единственную общую точку (точку их пересечения) $P(S_i)$.

Теперь рассмотрим вариант, при котором линия L и плоскость P пересекаются в единственной точке (см. Рис. 2).

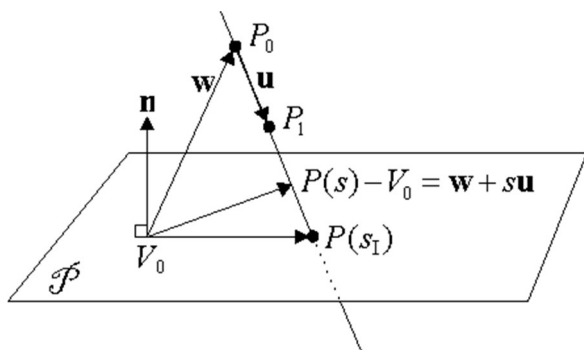


Рис. 2. Точка пересечения линии и плоскости

В рассматриваемой точке пересечения $P(S_i)$ вектор $P(S) - P_0 = W + SU$, где $W = P_0 - V_0$, перпендикулярен к нормали $n = (a, b, c)$, что эквивалентно следующему векторному произведению (3):

$$n(W + SU) = 0 \quad (3)$$

Тогда, исходя из вышеперечисленных фактов, имеет место выражение (4):

$$S_i = \frac{-n * w}{n * u} = \frac{n(V_0 - P_0)}{n(P_1 - P_0)} \quad (4)$$

Так как прямая L , заданная параметрическим уравнением (1), является отрезком от P_0 до P_1 , то необходимо что выполнялось условие:

$$0 \leq S_i \leq 1$$

Таким образом можно сказать, что при анализе возможных взаимных расположений полигонов двух различных объектов, необходимо применять алгоритмы, описанные выше до тех пор, пока не будет получено необходимое количество точек, а именно 4 точки, лежащие на одной прямой. Или, в противном случае, не обнаружится, что линии пересечения двух полигонов различных объектов не существует. И, чтобы определить существование упомянутых четырех точек, потребуется сопоставить все n полигонов первого объекта, со всеми m полигонами второго объекта, иначе говоря, придется совершить $n \times m$ проверок.

3 Оптимизация

Так как процесс поиска линии пересечения пары полигонов является независимой задачей, способной выполняться отдельно от всех остальных задач, то для повышения эффективности работы алгоритма в целом, целесообразным является его распараллеливание.

В задаче распараллеливания разделяемым ресурсом является массив отрезков, получающийся при пересечении двух полигонов. Каждый поток получает на вход примерно равное количество пар треугольников, которых он должен анализировать. После обработки очередной пары треугольников, в зависимости от результата, поток запрашивает монополярный доступ к разделенному ресурсу и помещает в выделенный ресурс результат. По окончании одного цикла поток проверяет очереди других, на наличие очередной пары треугольников. Если очереди всех других потоков оказываются пустыми, или работающих потоков кроме него не осталось, то он переходит в состояние синхронизации.

Для равномерного распределения m пар рассматриваемых треугольников на n потоков необходимо каждому из потоков присвоить $m \div n$ пар треугольников и первым $m \bmod n$ потокам добавить по одной паре.

4 Анализ результатов

По итогу проведения ряда экспериментов (см. Приложение 1) было установлено, что оптимальное количество потоков равно количеству логических ядер на используемой машине.

При превышении числа потоков выше количества логических ядер машины в начале работы алгоритма наблюдается незначительное сокращение времени поиска линии пересечения, но, в дальнейшем, эффективность падает. Подобное явление связано с квазипараллельными вычислениями, которые замедляют работу программы.

Главным результатом проведенного исследования является уменьшение времени обработки пропорционально количеству логических ядер машины. Полученная оптимизация является особенно ощутимой при рассмотрении высоко полигональных объектов.

Заключение

В данной статье подробно представлен единый алгоритм для окончательного поиска линии пересечения двух полигональных объектов по принципу треугольник-треугольник.

Представленный алгоритм не только обнаруживает факт пересечения моделей, но и возвращает, в качестве результата, линию наложения данных объектов. Помимо прочего, представлен один из лучших вариантов оптимизации данного метода, который сокращает время выполнения вычислений пропорционально количеству логических ядер машины.

Потенциально, данный алгоритм может быть усовершенствован путем сокращения числа сравнений полигонов. В будущем, улучшенный алгоритм будет полезен в таких областях как рендеринг в реальном времени, геометрическое моделирование и компьютерное зрение.

Список литературы

- Max J. Egenhofer, R. Franzosa, Point-Set topological Relations, International Journal of Geographical Information Systems 5(2), pp. 161 - 174, 1991.
- Oren Tropp, Ayellet Tal, Ilan Shimshoni, A fast triangle to triangle intersection test for collision detection, Computer Animation and Virtual Worlds, Vol17 (50), pp.527 - 535, 2006.
- G. Caumon, Collon - Drouaillet P, Le Carlier de Veslud C, Viseur S, Sausse J (2009) Surface - based 3D modeling of geological structures. Math Geosci 41:927–945, 2009.
- E.G. Houghton, Emmett R.F., Factor J.D. and Sabharwal C.L., Implementation of A Divide and Conquer Method for Surface Intersections; Computer Aided Geometric Design Vol.2, pp. 173 - 183, 1985.
- А. Д. Вайнштейн, “Построение минимального охватывающего параллелограмма”, Дискрет. матем., 2:4 (1990), 72–81
- А. В. Скворцов, Н. С. Мирза; Том. гос. ун-т, Фак. информатики. - Томск: Издательство Томского университета, 2006
- Chaman L. Sabharwal, Jennifer L. Leopold, Douglas McGeehan: Triangle-Triangle Intersection Determination and Classification to Support Qualitative Spatial Reasoning. Polibits 48: 13-22 (2013)

Приложение 1

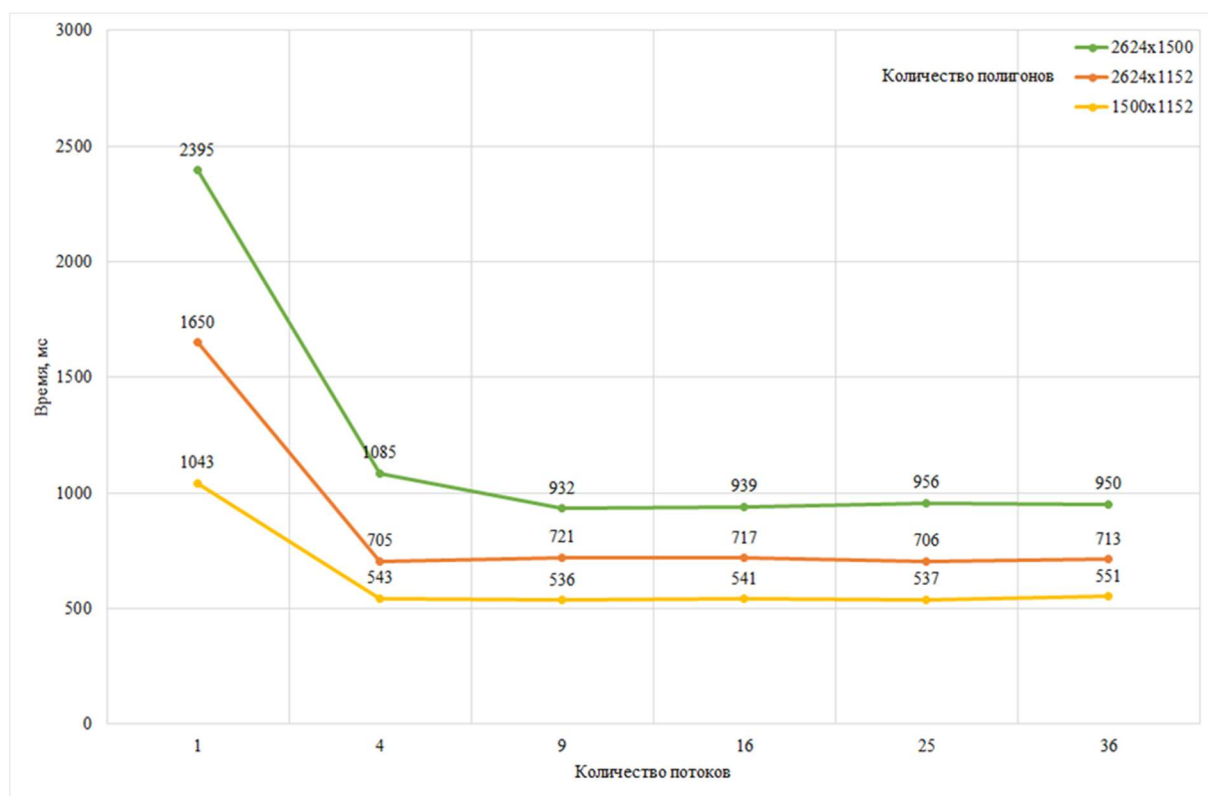


Рис. 3. Зависимость времени вычислений, от количества потоков при 4 логических ядрах

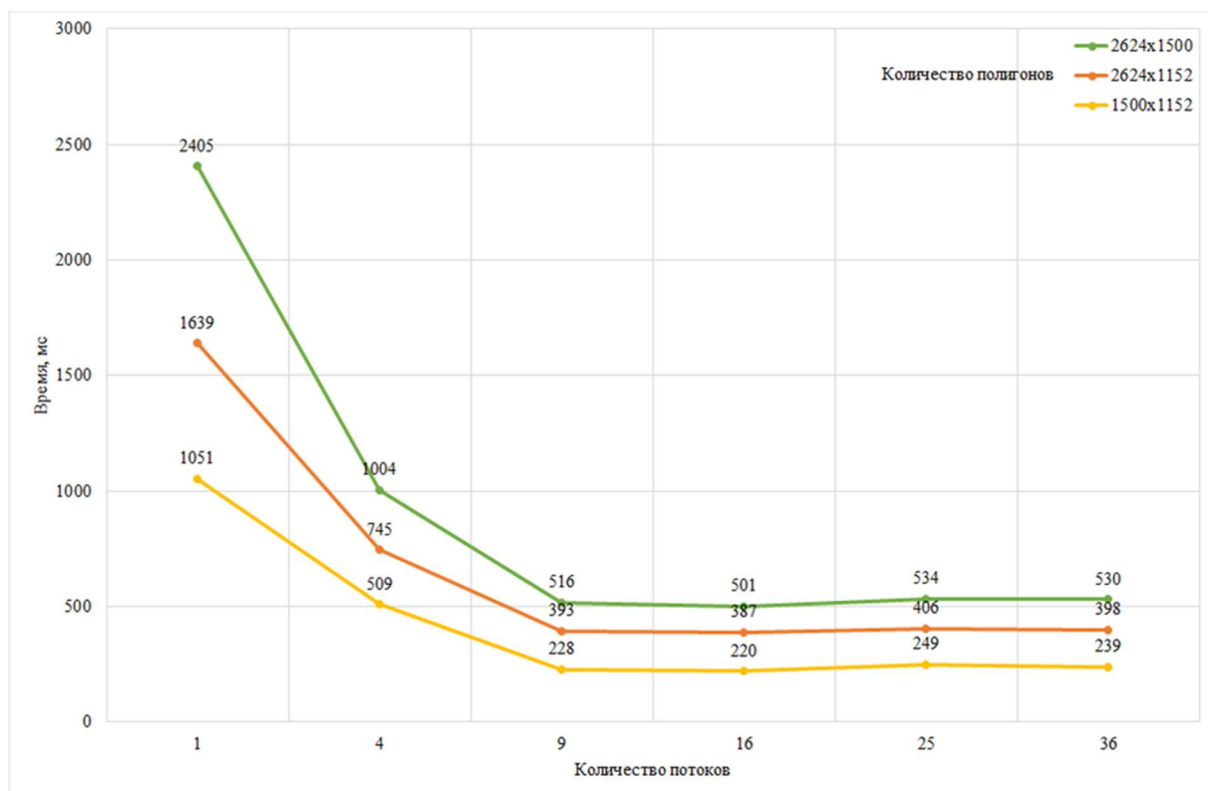


Рис. 4. Зависимость времени вычислений, от количества потоков при 8 логических ядрах