

# РЕАЛИЗАЦИЯ АЛГОРИТМОВ ТРИАНГУЛЯЦИИ ДЛЯ ТВЕРДОТЕЛЬНОГО МОДЕЛИРОВАНИЯ

Прорашнев В. Л.

Факультет информационных технологий и управления

Белорусского государственного университета информатики и радиоэлектроники

Минск, Республика Беларусь

E-mail: prorashnev.vsevolod@mail.ru

*В работе рассматриваются оптимизации алгоритма триангуляции для пакетов 3д-моделирования.*

## РАССМОТРЕНИЕ ВАРИАНТОВ ОПТИМИЗАЦИИ АЛГОРИТМА ТРИАНГУЛЯЦИИ

Процедура триангуляции в вычислительной геометрии - это разложение многоугольной фигуры  $P$  на некоторое множество треугольников, т.е. нахождение множества треугольников с попарно непересекающимися внутренними областями, объединением которых является  $P$ . Триангулировать любой выпуклый многоугольник за линейное время в верную триангуляцию триангуляционно, добавляя диагонали от одной вершины ко всем остальным вершинам. Общее количество способов триангулировать выпуклый  $n$ -угольник непересекающимися диагоналями - это  $(n-2)$ -е число Каталана, которое можно вычислить по формуле (1), открытой Леонардом Эйлером.

$$C_{n-2} = \frac{n(n+1) \dots (2n-4)}{(n-2)!}. \quad (1)$$

Согласно данной формуле на примере выпуклого семиугольника (7-стороннего выпуклого многоугольника) имеем 42 возможных триангуляции (рисунок 1). Это количество задается 5-м каталанским числом.

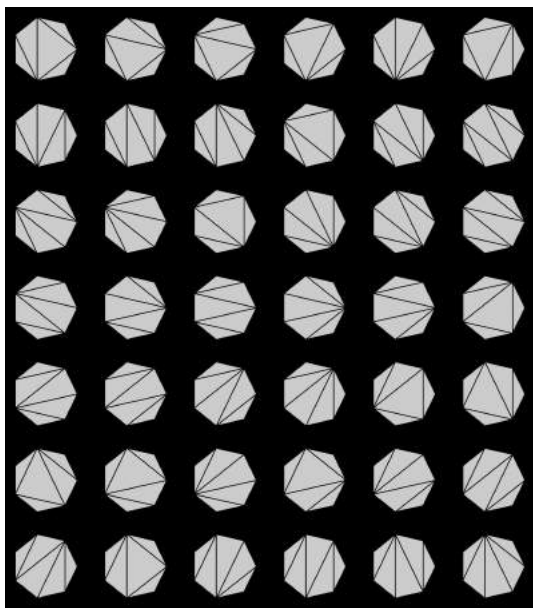


Рис. 1 – 42 варианта триангуляции семиугольника

Однако, верная триангуляция перестает быть актуальной при рассмотрении более слож-

ных многоугольников, не удовлетворяющих требованию выпуклости. Для такого случая существуют методы «обрезания ушей» и «монотонной триангуляции». [1]

Метод обрезания ушей основывается на том, что любой простой многоугольник с минимум 4 вершинами без отверстий имеет как минимум два «уха» – треугольники, две стороны которых являются краями многоугольника, а третья сторона находится полностью внутри него. Алгоритм состоит в том, чтобы найти такое ухо, удалить его из многоугольника (в результате чего получается новый многоугольник, который все еще удовлетворяет условиям) и повторять, пока не останется только один треугольник. Этот алгоритм прост в реализации, но работает медленнее, чем некоторые другие алгоритмы, и он работает только на многоугольниках без отверстий. Эффективный алгоритм обрезания ушей был найден Хоссамом Эль Гинди, Хейзел Эверетт и Годфридом Туссенем.

Монотонная триангуляция многоугольников заключается в разбиении многоугольника на монотонные многоугольники. Простой многоугольник является монотонным относительно прямой  $L$ , если любая прямая, ортогональная к  $L$ , пересекает многоугольник не более двух раз. Монотонный многоугольник может быть разбит на две монотонные цепи. Многоугольник, монотонный относительно оси  $y$ , называется  $y$ -монотонным. Предполагая, что данный многоугольник является  $y$ -монотонным, алгоритм начинается с прохождения по одной цепи многоугольника сверху вниз, добавляя диагонали при каждой возможности. Данный алгоритм может быть эффективно применен к любому монотонному многоугольнику. Если многоугольник не является монотонным, его можно разбить на монотонные подполигоны, используя подход с использованием линий развертки. Алгоритм не требует, чтобы многоугольник был простым, поэтому его можно применять к многоугольникам с отверстиями.

В рамках компьютерной графики триангуляция является неотъемлемым этапом подготовки полигональных объектов к работе, поскольку 3д пакеты, использующие полигональные модели, в конечном итоге предназначены для работы с треугольниками. При попытке использо-

вания в работе многоугольников с большим количеством вершин велика вероятность неверного отображения визуализации объекта. Исключением для этой процедуры могут являться четырехугольные многоугольники за счет меньшей склонности к созданию ошибок отображения в связи с малым количеством вершин и вариантов триангуляции. Однако, в трехмерном пространстве триангуляция многоугольника не дает гарантий правильного его отображения. Наиболее актуальна данная проблематика для сферы анимации, когда при деформации триангулированного многоугольника его вершины перемещаются неравномерно, тем самым создавая новые, более сложные формы из образованных во время триангуляции диагоналей. Анализируя условия возникновения некорректного отображения объекта при деформации триангулированного многоугольника, можно прийти к выводу, одной из основных причин является сонаправленность ребер на внутренней площади многоугольника, что обусловлено недопустимостью пересечения внутренних граней по определению триангуляции. Таким образом, для снижения вероятности ошибок визуализации триангулированных многоугольников требуется модификация алгоритма триангуляции, не ограниченная требованиями о непересечении внутренних граней. Логичным шагом для получения данной модификации будет добавление дополнительных вершин в центр периметра многоугольника. Поскольку в общем случае неизвестно, достаточно ли будет данной операции для избежания ошибок визуализации, следует добавить итеративность в этап добавления вершин. На первой итерации одна дополнительная вершина помещается в центр массы многоугольника. Алгоритм для нахождения центра массы объекта в двух и трех измерениях уже реализован для всех пакетов трёхмерного проектирования, работающих с полигональными объектами, и подходит для применения в рамках рассматриваемой модификации алгоритма триангуляции. На второй итерации рассматриваются отсекаемые прилегающими к дополнительной вершине ребрами участки многоугольника и для участка с наибольшей площадью находится центр массы и создается новая дополнительная вершина. Последующие итерации повторяют процедуру, рассматривая отсекаемые набором ребер, прилегающих к дополнительным вершинам, участки многоугольника. На рисунках 2-3 представлено сравнение результата триангуляции с результатом двух итераций модифицированного алгоритма на примере случайно сгенерированного  $n$ -угольника, где  $n$  равно 32.

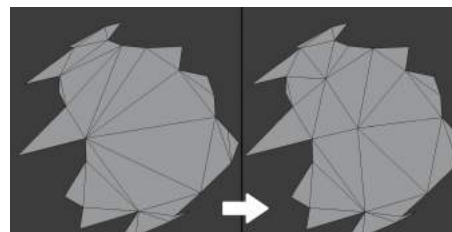


Рис. 2 – Пример оптимизации алгоритма триангуляции

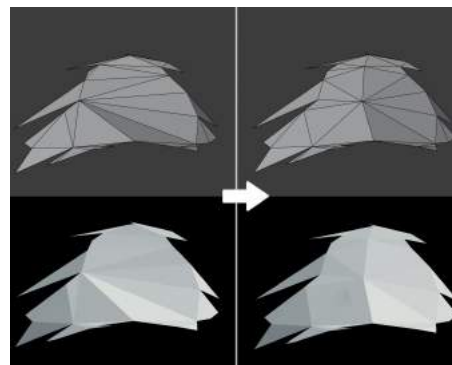


Рис. 3 – Пример визуализации многоугольника до модификации алгоритма триангуляции и после

В рассмотренном примере два одинаковых многоугольника подвергались одинаковой деформации, - сгиб по оси  $y$ . Добавление всего двух новых вершин помогло избежать потери формы и, как следствие, ошибок визуализации после триангуляции. Однако, в примере рассматривалась деформация двухмерной фигуры, поскольку все вершины исходного многогранника находились в одной плоскости. Рассмотрим пример с трехмерной фигурой, часть которой заменим сложным  $n$ -угольником.

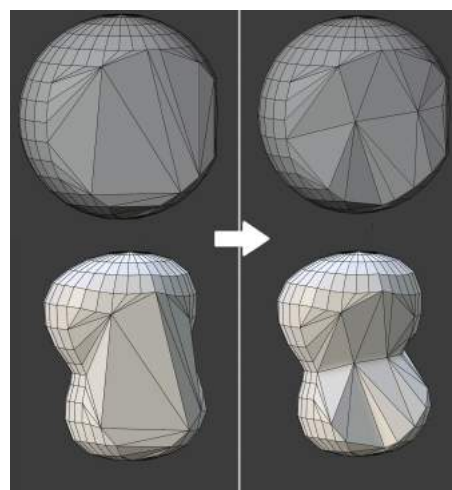


Рис. 4 – Пример визуализации многоугольника до модификации алгоритма триангуляции и после

## I. СПИСОК ЛИТЕРАТУРЫ

1. Proceedings of the 2012 International Conference on Information Technology and Software Engineering / Ed. J. Wei Lu, Guoqiang Cai, Weibin Liu // Springer-Verlag Berlin Heidelberg. – 2013. – P.980-984.