Восстановление топологической информации из мультитриангуляционных моделей поверхностей

Н.С. Мирза, А.В. Скворцов
Томский государственный университет,
Томск, Россия
{mirza, skv}@indorsoft.ru

Аннотация

В данной работе автором предлагается новый эффективный способ восстановления топологической информации из мультитриангуляционных моделей, который позволяет быстро устанавливать связи треугольников в извлечённой поверхности.

Ключевые слова: *Мультитриангуляция, триангуляция Делоне, вычислительная геометрия, 2,5-мерная поверхность, мультитриангуляционная модель, топологическая информация.*

1. ВВЕДЕНИЕ

На сегодняшний день задачи построения и обработки цифровых моделей поверхностей находят широкое применение в графических системах. Для построения модели поверхности, как правило, используют триангуляцию Делоне, однако для эффективной работы с триангуляцией, построенной по огромному массиву входных данных необходимо иметь мощное аппаратное обеспечение или использовать специальные алгоритмы упрощения. Но использование алгоритмов упрощения изначально подразумевает необходимость нахождения некоторого компромисса между быстродействием обработки поверхности и качеством получаемых результатов. Данная проблема решается с помощью использования *мультитриангуляции* (*МТ*) – особой структуры, состоящей из фрагментов триангуляции различных уровней детализации [1]. Детализация мультитриангуляционной модели устанавливается согласно некоторому критерию, который определяется видом решаемой задачи, поэтому, подобрав подходящий критерий, можно получить модель, скорость обработки и качество которой будут очень высоки.

В нас тоящее время существует достаточно много алгоритмов, основанных на МТ, однако, до сих пор не предложено эффективного способа восстановления из МТ топологической информации, которая необходима для решения многих задач, в том числе для расчета изолиний, изоклинов, экспозиций склонов, объёмов земляных работ и др.

2. ОБЗОР ПРЕДМЕТНОЙ ОБЛАСТИ

2.1 Основные понятия

Прежде всего, отметим, что в данной работе под термином «поверхность» будем понимать однозначную функцию высот от планового положения точек. Такие поверхности ещё называют *2,5-мерными*, чтобы отличить это понятие от значения более общего понятия *3-мерная* поверхность, принятого в математике.

Для строгого определения мультитриангуляции необходимо ввести ряд терминов.

Определение. *Триангуляция* – планарный граф, все конечные грани которого являются треугольниками [1].

Определение. Триангуляция *удовлетворяет условию Делоне*, если в окружность, описанную около любого треугольника, не попадает ни одна из заданных точек триангуляции.

Определение. Триангуляция называется *триангуляцией Делоне*, если она является выпуклой и удовлетворяет условию Делоне (рис. 1).

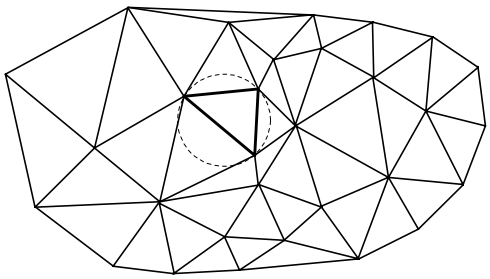

Рис. 1. Триангуляция Делоне

Определение. Пусть заданы две триангуляции T и T_i такие, что T_i занимает область меньшую, нежели T. Тогда T и T_i называются *совместимыми* [1], если в T существует подмножество треугольников $\overline{T}$, занимающих в точности ту же область, что и T_i и при этом $\overline{T}$ является триангуляцией. В таком случае, T_i называется *фрагментом* (относительно T), а $\overline{T}$ *– покрываемой областью* T_i.

Определение. T_i называется *минимально* совместимым с T, если в T_i нет подтриангуляции совместимой с T.

Определение. *Локальной модификацией* T через фрагмент T_i является операция замены треугольников $\overline{T}$ треугольниками T_i и обозначается $T \oplus T_i$.

Определение. Если все T_i совместимы с T, то последовательность преобразований $t = (T_0,...,T_n)$ называется *совместимой последовательностью триангуляций*.

Определение. В терминах, определённых выше, ***мультитриангуляция*** – направленный граф без циклов, вершинами которого являются элементы совместимой последовательности триангуляций [1] (рис.1).

Определение. *Корнем* графа является фрагмент с максимальным (минимальным) уровнем детализации, а *стоком* – с минимальным (максимальным) для МТ, построенной с помощью алгоритмов упрощения (детализации).

Мультитриангуляцию обычно представляют в виде следующих структур данных [0]:

1. *Список фрагментов*: каждый фрагмент содержит список составляющих его треугольников и список указателей на треугольники покрываемой части.

2. *Список треугольников*: каждый треугольник содержит ссылки на три образующих его узла, ссылки на фрагмент, где он содержится (верхний фрагмент) и на фрагмент, который содержит его в покрываемой части (нижний фрагмент).

3. *Список вершин*: каждая вершина содержит необходимый набор геометрических данных, таких как: координаты (*X*, *Y*, *Z*), нормаль, и т.п.

Существует несколько алгоритмов построения МТ, для примера далее приводится алгоритм, предложенный в [2], который начинает работу с триангуляции Делоне максимальной детализации, и путём последовательных шагов упрощения строит МТ.

Алгоритм построения МТ

Входные данные: триангуляция Делоне.

Выходные данные: МТ.

Структура алгоритма:

Шаг 1. На базе исходной триангуляции создаётся первый фрагмент МТ, являющийся корнем графа МТ.

Шаг 2. Находится множество узлов триангуляции *P*, содержащее максимальное число не связанных ребром узлов.

Шаг 3. Если множество *P* пусто, то МТ построена, и алгоритм завершает работу.

Шаг 4. Производится удаление всех узлов из P. При удалении каждого очередного узла, создаётся новый фрагмент МТ из треугольников, содержащих удаляемую точку. После этого устанавливается связь между треугольниками созданного фрагмента и новыми треугольниками триангуляции, образованными после удаления.

Шаг 5. Осуществляется новый этап упрощения, т.е. переход на Шаг 2.

Конец алгоритма.

Пример построения МТ описанным алгоритмом изображён на рис.1.

Трудоёмкость данного алгоритма $O(N)$, где N – число узлов в триангуляции [2].

Для эффективной работы с МТ в ней должна быть представлена следующая информация [3]:

1. Геометрическая информация – узлы и треугольники МТ.

2. Топологическая информация – связи треугольников, т.е. для каждого треугольника определяется информация о соседних треугольниках (относительно ребра или вершины).

3. Пространственная информация – граф МТ, который определяет взаимозависимости треугольников, представляющих различную детализацию поверхности.

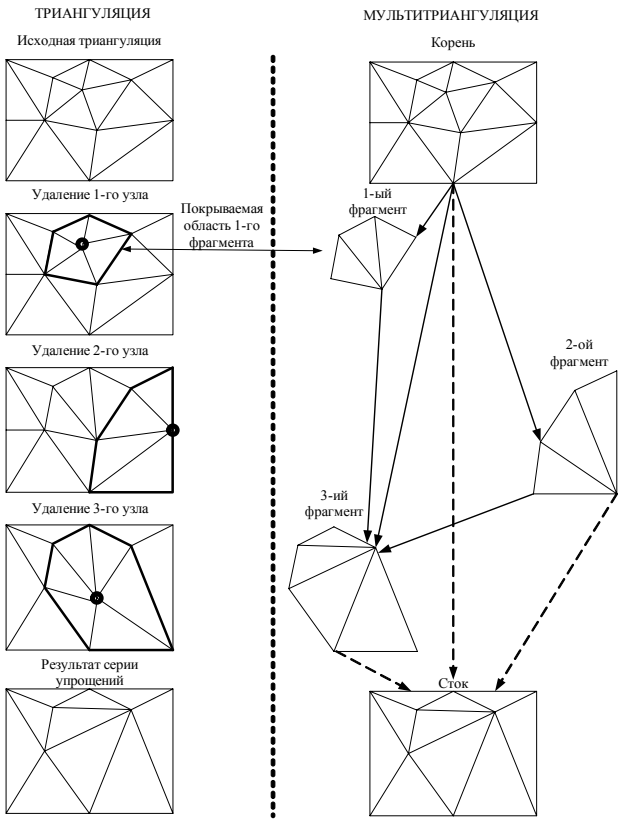

Рис. 2. Пример МТ

Для представления и хранения в МТ геометрической и пространственной информации предложено множество структур данных, позволяющих эффективно работать с МТ. Однако до сих пор не предложено достаточно эффективного способа извлечения из МТ топологической информации. Основная проблема данной задачи заключается в том, что каждый треугольник может по одному ребру соседствовать с множеством треугольников в зависимости от детализации извлекаемой поверхности (рис.2).

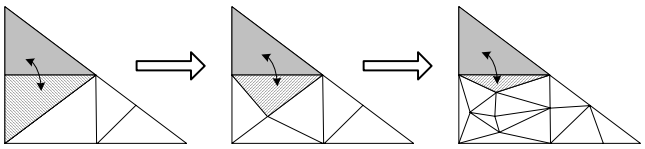

Рис. 3. Смена соседних треугольников при увеличении детализации поверхности

Следовательно, получение топологической информации из МТ возможно только для треугольников извлечённой поверхности, так как для всех остальных треугольников существует целый ряд альтернативных соседей.

2.2 Существующие решения

На сегодняшний день задача эффективного нахождения соседей для каждого треугольника МТ занимает умы многих учёных. Далее предлагается обзор различных подходов к восстановлению топологической информации в МТ, предложенных за последние годы.

Если в классической МТ в явном виде отсутствует топологическая информация, то самый эффективный алгоритм восстановления связей треугольников будет работать с трудоёмкостью $O(N_R \log N_R)$, где N_R – число треугольников в извлечённой поверхности [3].

В работах Л. де Флориани, Е. Пуппо и П. Магилло предлагается структура МТ, для каждого треугольника которой необходимо хранить список его возможных соседей (смежные треугольники с каждого этапа построения) [3].

Такая структура МТ значительно увеличивает расход оперативной памяти. К тому же, для того чтобы получить соседей для каждого треугольника, необходимо каждый раз просматривать список его потенциальных соседей и выбирать тот треугольник, который участвует в формировании извлекаемой поверхности. Поэтому для данной структуры алгоритм восстановления связей треугольников будет работать с трудоёмкостью $O(N_R \log A)$, где A – максимальное число соседей. В наихудшем случае $A = O(N_R)$ [3].

В работах В. Эванса, П. Линдстрома, Е. Пуппо и др. предлагается некоторая модификация классической МТ – *RTIN-Иерархия*, которая применяется для моделирования 2,5-мерных поверхностей [4].

RTIN-Иерархия позволяет очень компактно хранить геометрические данные и извлекать геометрическую и топологическую информацию для каждого треугольника за время $O(1)$. В основе этой структуры лежит разбиение поверхности на равнобедренные «правоугольные» треугольники (*RTIN*). Это разбиение занимает промежуточное положение между регулярной (*Sud-grid*) и нерегулярной (*TIN*) моделью поверхности.

RTIN-Иерархия представляет собой бинарное дерево, построение которого начинается с самой грубой аппроксимации поверхности, состоящей из двух треугольников, каждый из которых по заданному алгоритму рекурсивно делится на несколько «правоугольных» треугольников. При этом каждому треугольнику ставится в соответствие некоторое число, кодирующий его положение в дереве. Таким образом, зная этот двоичный код, имеется возможность быстро восстановить геометрическую и топологическую информацию поверхности.

Однако RTIN-Иерархию достаточно сложно реализовать на практике, и для решения некоторых задач, связанных с моделированием 2,5–мерных поверхностей, применяться она не может. Так, например, для задач, требующих точного учёта микрорельефа, необходимо работать с моделью поверхности с заданными узлами и структурными линиями. И в этом случае этот алгоритм неприменим.

Поэтому необходимо разработать новый эффективный способ извлечения топологической информации, работающий для классической МТ. При этом, видимо, необходимо будет дополнить структуры данных МТ некоторыми дополнительными полями.

3. ПРЕДЛАГАЕМОЕ РЕШЕНИЕ

3.1 Изменение структуры МТ

В данной работе автором предлагается решения проблемы соседства треугольника со многими другими треугольниками по одному ребру (см. рис. 3) путём следующего изменения структуры МТ.

1. Хранение для каждого треугольника ссылок на 3 самых вероятных соседних треугольника.

Идея предлагаемого автором алгоритма извлечения топологической информации основана на том, что построение МТ осуществляется на базе топологически связанной триангуляции. Соответственно соседи треугольников в триангуляции являются наиболее *вероятными соседями* и в извлекаемой из МТ поверхности. Поэтому автором предлагается для каждого треугольника хранить 3 ссылки на те треугольники, которые были соседними в триангуляции в момент его создания.

Тем самым предоставляется возможность восстановления связей треугольников в извлечённой поверхности для *типичного случая*, когда один из треугольников поверхности ссылается на своего актуального соседа, а другой нет (рис. 4).

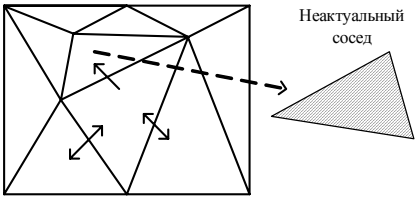

Рис. 4. Типичный случай организации ссылок на соседние треугольники в извлечённой поверхности

Таким образом, для типичного случая восстановления топологической информации всего лишь достаточно просмотреть весь список треугольников извлечённой поверхности и установить всем треугольникам взаимные ссылки.

Однако для некоторых треугольников извлечённой поверхности возможен *вырожденный случай* организации ссылок, когда соседние треугольники в извлечённой поверхности ссылаются на треугольники, не участвующие в формировании поверхности (на *неактуальных соседей*). Это связано с тем, что соседними оказываются треугольники, которые не были соседними ни на одном этапе упрощения исходной триангуляции (рис. 5).

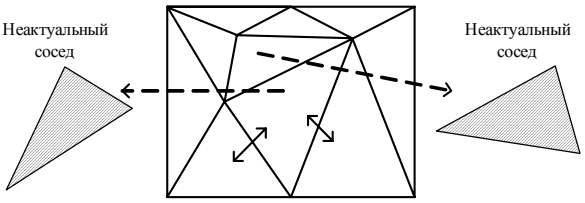

Рис. 5. Вырожденный случай организации ссылок на соседние треугольники в извлечённой поверхности

Для учёта такого вырожденного случая автором предлагается следующее решение.

2. Хранение для каждого треугольника ссылок на верхние и нижний треугольники.

Для каждого треугольника необходимо хранить ссылки на *нижний* и *верхние* треугольники (рис. 6). *Верхний треугольник* детализирует треугольник по заданному ребру, а *нижний* упрощает. Заметим, что детализировать треугольник можно по каждому из трёх рёбер, а вот упрощение возможно лишь по одному ребру, так как два других будут удалены вместе с удаляемым узлом триангуляции (см. рис. 6).

Использование верхних и нижних треугольников предоставляет возможность найти всю цепочку треугольников МТ, смежных заданному ребру, а следовательно определить всех возможных треугольников-соседей данного ребра.

Таким образом, при возникновении вырожденного случая, необходимо найти соответствующий верхний или нижний треугольник *не актуального* соседа, принадлежащий извлечённой поверхности – это и будет искомый актуальный сосед.

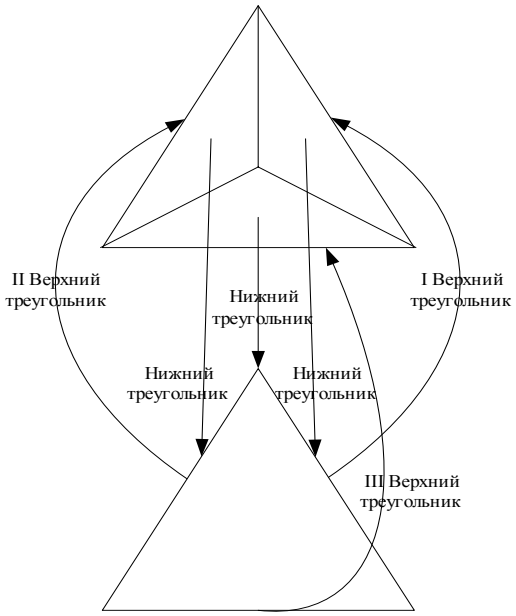

Рис. 6. Пример верхних и нижних треугольников

Для определения ссылок на соседние треугольники автором предлагается следующий алгоритм, реализующий решение указанных проблем.

Алгоритм определения самых вероятных соседей

Выходные данные: МТ, для каждого треугольника которой заполнены ссылки на самые вероятные соседние треугольники.

Описание алгоритма:

Алгоритм представляет собой две процедуры. Первая процедура работает во время создания нового треугольника МТ. Она заполняет ссылки на самых вероятных его соседей. Однако для некоторого треугольника МТ соседним может оказаться треугольник, который пока ещё отсутствует в МТ. Поэтому предлагается хранить вспомогательный список таких «отсутствующих» треугольников, в каждом элементе которого будет записано пустое значение, пока треугольник отсутствует в МТ. По мере создания треугольников список будет заполняться ссылками на уже существующие треугольники. Следовательно, если необходимо установить ссылку на соседний треугольник, который отсутствует в МТ, то в списке отсутствующих треугольников создаётся новый элемент с пустым значением, а в качестве соседнего треугольника устанавливается ссылка на номер элемента в этом списке.

Вторая же процедура работает после завершения построения МТ и устанавливает всем треугольникам, ссылающимся на некоторые ранее несуществовавшие треугольники, ссылки на уже созданные треугольники, хранящиеся в списке отсутствующих треугольников.

Структуры данных:

1. Треугольник триангуляции:

```
TriangulationTriangle = class
…
// Самые вероятные соседние треугольники
AdjTr: array [1..3] of TriangulationTriangle;
// Соответствующий треугольник МТ
MTTr: MTTriangle;
// Номер в списке отсутствующих треугольников
MTTrIndex: integer;
end;
```

2. Треугольник МТ:

```
MTTriangle = class
…
// Соседние треугольники
AdjTr: array [1..3] of MTTraingle;
// Индексы соседних треугольников в TriangleInds
AdjTrInds: array [1..3] of integer;
// Нижний треугольник
LowTriangle: MTTriangle;
// Верхние треугольники
UpTriangles: array [1..3] of MTTraingle;
end;
```

Процедуры и функции:

1. `GetNeighIndex(T, Neighbour: TriangulationTriangle): integer` и `GetMTNeighIndex(T, Neighbour: MTTriangle): integer` – определяет номер треугольника **`T`** в качестве соседа `Neighbour` – соответственно для треугольников триангуляции и мультитриангуляции (рис. 7). Отметим, что нумерация узлов в треугольнике производится по часовой стрелке, при этом `i`-й соседний треугольник лежит напротив `i`-й вершины.

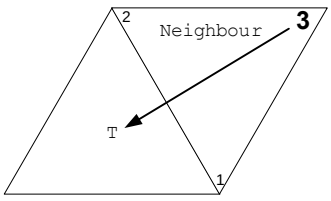

Рис. 7. Пример работы функций `GetNeighIndex/GetMTNeighIndex`

2. `CreateNewMTTriangle: MTTriangle` – создаёт в МТ новый треугольник и возвращает на него ссылку.

3. `IsEmpty(Item): boolean` – проверка на пустое значение.

1. Алгоритм создания треугольника МТ.

Входные данные:

1. Треугольник `Tr` исходной триангуляции, из которого создаётся треугольник МТ.

2. Список отсутствующих треугольников `TriangleInds`, для которого определены стандартные операции (`Add`, `Count`, `[]`).

Выходные данные: треугольник МТ с установленными ссылками на соседей.

Структура алгоритма:

```
NewTr:=CreateNewMTTriangle;
for k:=1 to 3 do
  if not IsEmpty(Tr.AdjTr[k]) then
  begin
    Neighbour:=Tr.AdjTr[k].MTTr;
    if IsEmpty(Neighbour) then
    begin
      if Tr.AdjTr [k].MTTrIndex<0 then
      begin
        Tr.AdjTr[k].MTTrIndex:=TriangleInds.Count;
        TriangleInds.Add(Empty);
      end;
      NewT.TrInds[k]:=Tr.AdjTr[k].MTTrIndex;
    end;
    NewT.AdjTr[k]:=Neighbour;
  end
  else
    NewT.AdjTr[k]:=Empty;
if Tr.MTTrIndex>=0 then
  TriangleInds[T.MTTrIndex]:=NewTr;
Tr.MTTr:=NewT;
```

Конец алгоритма.

2. Алгоритм, работающий после создания МТ.

Структура алгоритма:

```
for i:=1 to MTTrCount do
  for j:=1 to 3 do
  begin
    index:=MTTr[i].AdjTrInds[j];
    if index>=0 then begin
      MTTr [i].AdjTr[j]:=TriangleInds[index];
      n:=GetMTNeighIndex(MTTr[i], TrInds[index]);
      TrInds[index].AdjTr[n]:=TriangleInds[index];
      TriangleInds[index].AdjTrInds[n]:=-1;
    end;
  end;
```

Конец алгоритма.

Данный алгоритм очевидно работает с трудоёмкостью $O(N)$, где N – число треугольников в МТ. Это следует из простой структуры алгоритма, содержащего только одинарные циклы. Таким образом, общая трудоёмкость алгоритма построения МТ не ухудшается.

3.2 Извлечение топологической информации

Чтобы восстановить из предложенной МТ связи всех треугольников уже извлечённой поверхности, автором предлагается следующий алгоритм.

Алгоритм восстановления топологической информации

Входные данные: Список треугольников извлечённой из МТ поверхности.

Выходные данные: топологически связанная поверхность.

Процедуры и функции:

1. `IsExtracted (T: MTTriangle): boolean` – принадлежит ли `T` извлечённой поверхности.

2. `SaveAdjTrInfo(T: MTTriangle; i: integer)` – сохраняет ссылку на **i**-ого соседа `T` во временном буфере.

3. `FindLow(Up)Triangle(TT, T: MTTriangle): MTTriangle` – находит нижний (верхний) треугольник `TT` смежный с `T`, который принадлежит извлечённой поверхности.

Структура алгоритма:

Шаг 1. Установление ссылок на соседние треугольники

```
for i:=1 to ExtractedTrCount do
begin
  T:=MTTr[i];
  for j:=1 to 3 do
  begin
    TT:=T.AdjTr[j];
    if not IsEmpty(TT) and IsExtracted (TT) then
    begin
      index:=GetMTTrNeighIndex(T, TT);
      SaveAdjTrInfo(TT, index);
      TT.AdjTr[index]:=T;
    end;
  end;
end;

for i:=1 to ExtractedTrCount do
begin
  T:=MTTr[i];
  for j:=1 to 3 do
  begin
    TT:=T.AdjTr[j];
    if not IsEmpty(TT) and not IsExtracted (TT) then
    begin
      TT:=FindLowTriangle(TT, T);
      if IsEmpty(TT) then
        TT:=T.AdjTr[j];
      TT:=FindUpTriangle(TT, T);
      SaveAdjTrInfo(T, j);
      T.AdjTr[j]:=TT;
      SaveAdjTrInfo(TT, index);
      TT.AdjTr[index]:=T;
    end;
  end;
end;
```

Шаг 2. Восстановление из временного буфера исходных ссылок на соседей для треугольников МТ.

Конец алгоритма.

Исходя из структуры алгоритма, можно сделать вывод, что его трудоёмкость составляет $O(N_R + \overline{N_R} \cdot \log N_R)$, где $\overline{N_R}$ – число треугольников в извлечённой поверхности, которые не были соседними в исходной триангуляции (т.е. для них необходимо искать соответствующий верхний или нижний треугольник).

Замечание. Автору не удалось получить аналитическую оценку параметра $\overline{N_R}$. Однако, согласно проведённым автором экспериментальным исследованиям (см. ниже п. 4.2) $\overline{N_R} = O(N_R / c^{N_R})$, где c – некоторая константа. Следовательно, трудоёмкость предложенного алгоритма составляет в среднем $O(N_R + N_R / c^{N_R} \cdot \log N_R) = O(N_R)$.

Таким образом, предложенное решение восстановления топологической информации превосходит существующие аналоги по быстродействию.

Следует заметить, что если для каждого треугольника МТ хранится ссылка на его нижний треугольник, то нет необходимости хранить ссылку на его нижний фрагмент. К тому же для каждого треугольника МТ верхних треугольников может быть не более трёх. Таким образом, расход дополнительной оперативной памяти для предложенной МТ (по сравнению с классической структурой) составляет $3 \cdot 3 \cdot 4 \cdot N = 36 \cdot N$ байт, что значительно меньше по сравнению с аналогичными решениями других авторов.

4. ЭКСПЕРИМЕНТАЛЬНОЕ МОДЕЛИРОВАНИЕ

4.1 Условия эксперимента

Для апробации предложенных в данной работе алгоритмов и оценки некоторых их параметров автором была выполнена их программная реализация, а затем было произведено экспериментальное моделирование работы алгоритмов на различных наборах данных, отличающихся типом закона распределения исходных точек и их количеством.

Программирование алгоритмов было выполнено на языке Pascal в среде Borland Delphi, а само моделирование было произведено на компьютере с процессором Intel Pentium 4, 2400 МГц и оперативной памятью 512 Мб.

4.2 Оценка числа вырожденных случаев

В ходе эксперимента была произведена оценка числа вырожденных случаев $\overline{N_R}$, т.е. числа треугольников, для которых необходимо осуществлять поиск верхних или нижних треугольников. На рис. 8 изображена зависимость $k = N_R / \overline{N_R}$ от N_R.

Таким образом, исходя из экспериментальных результатов (см. рис. 8) можно сделать предположение о том, что в среднем $\overline{N_R} = O(N_R / c^{N_R})$, где c –некоторая константа.

Вывод: $\overline{N_R}$ – величина большего порядка малости, чем N_R.

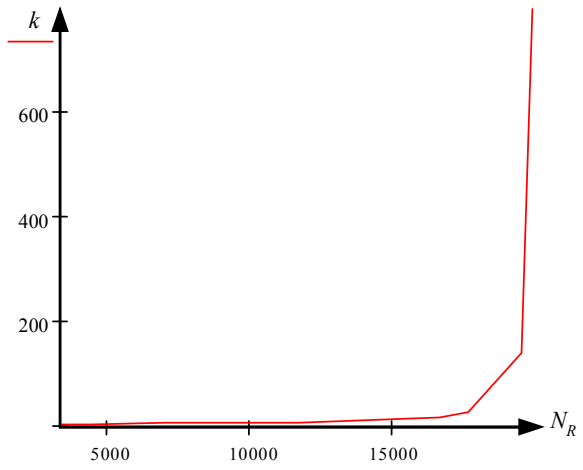

Рис. 8. Зависимость $k = N_R / \overline{N_R}$ от числа треугольников в извлечённой поверхности

4.3 Быстродействие алгоритма

Полученные результаты по оценке быстродействия алгоритма представлены в табл.7.

На рис. 9 представлен соответствующий график зависимости времени работы алгоритма от числа треугольников в извлечённой поверхности.

Таблица 7. Время работы алгоритма восстановления топологии из К-МТ , мс

N_R	3329	4370	6995	11691	16592	17576	19460
Время работы алгоритма, мс	330	338	42	46	50	53	56

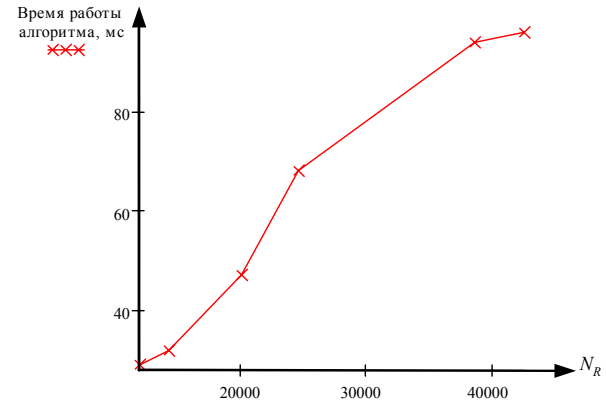

Рис. 9. График зависимости времени работы алгоритма от числа треугольников в извлечённой поверхности

5. ЗАКЛЮЧЕНИЕ

В данной работе автором предлагается новый способ восстановления топологической информации для мультитриангуляционных моделей поверхности, основанный на изменении структуры MT, который отличается от аналогичных решений существенно меньшим расходом оперативной памяти и более высоким быстродействием.

6. БИБЛИОГРАФИЯ

[1] Скворцов А.В. Триангуляция Делоне и её применение. – Томск: Изд-во Том. ун-та, 2002. – 128 с.

[2] E. Puppo Variable resolution triangulations // Computational Geometry. 1998. Vol. 11. P. 219–238.

[3] M. de Berg, K. Dobrindt. On levels of detail in terrains // Proceedings 11th ACM Symposium on Computational Geometry. 1995. P. C26–C27.

[4] L. de Floriani, P. Magillo, E. Puppo, M. Bertolotto Variable resolution operators on a multiresolution terrain model // 4th ACM Workshop on Advances in Geographic Information Systems. 1996. P. 123–130.

[5] W. Evans, D. Kirkpatrick, G. Townsend Right-triangulated irregular networks // Algorithmica. 2001. P. 264-286.

Об авторах

Мирза Наталия Сергеевна – аспирант Томского государственного университета, факультет прикладной математики и кибернетики
Адрес: Томск, 634050, пр-т Ленина, 36, ТГУ.
Телефон: 8(3823) 99-94-81
E-mail: mirza@indorsoft.ru

Скворцов Алексей Владимирович – доцент, д.т.н., профессор Томского государственного университета.
Адрес: Томск, 634050, пр-т Ленина, 36, ТГУ.
Телефон: 8(3822) 65-13-86
E-mail: skv@indorsoft.ru.

Reconstruction of topological information from Multiresolutional models

Abstract

Here is described a new efficient method of topological information reconstruction from Multiresolutional models, which allows fast triangles neighbor finding of extracted surface.

Keywords: *Multi-Triangulation, Delaunay triangulation, Computational geometry, Multiresolution modeling, Topological information*.

About the author(s)

Natalia Mirza is a post-graduate student at Tomsk State University, Department Of Computer Science. Her contact e-mail is mirza@indorsoft.ru.

Alexey Skvortsov is a professor at Tomsk State University, Department of Theoretical Basis of Computer Science. His contact email is skv@indorsoft.ru.