УДК 681.3

СГЛАЖИВАНИЕ ТРИАНГУЛИРОВАННОЙ ПОВЕРХНОСТИ

Н.А. Тюкачев

Предлагается алгоритм сглаживания триангулированной поверхности, основанный на введении дополнительного разбиения каждого треугольника на несколько новых треугольников, вершины которых лежат на бикубической поверхности Безье

Ключевые слова: геометрические алгоритмы, сглаживание, полиномы Безье

[1] При проведении геологоразведочных работ бурят сотни скважин и берут сотни километров проб, определяя, в частности, литологию и стратиграфию слоев породы. Скважины бурят достаточно далеко друг от друга. Линии скважин находятся на расстоянии метров 400-500 друг от друга. В линиях расстояние между ними примерно такое же.

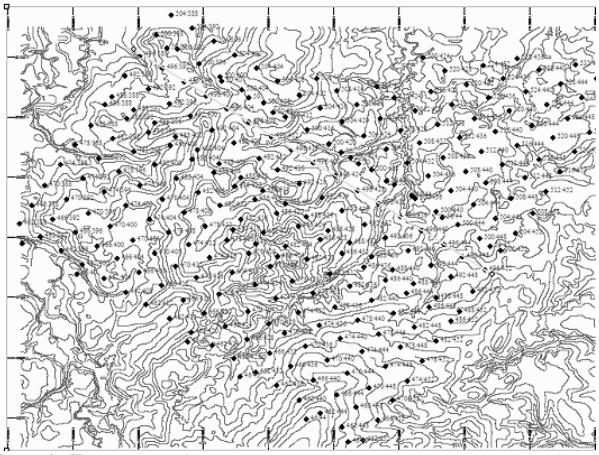

Рис. 1. Точками обозначены скважины

Триангуляция поверхностей слоев, построенная на скважинах, образует редкую сетку треугольников (рис. 2).

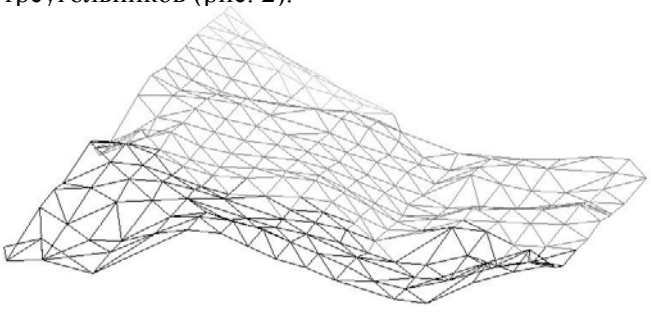

Рис. 2. Триангулированная по скважинам поверхность слоя

Поэтому возникает следующая задача: построить интерполирующую поверхность, которую можно использовать для введения дополнительного разбиения каждого треугольника на несколько новых треугольников. Новые узловые точки не должны лежать в плоскости треугольника и использоваться для сглаживания поверхности (рис. 3).

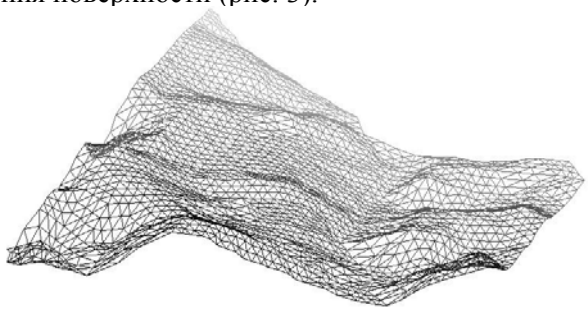

Рис. 3. Поверхность, сглаженная добавленными треугольниками

Интерполирующая поверхность необходима для расчета сечений и значений z в требуемых точках области определения, а также для визуализации. Система пространственных треугольников позволяет очень просто решить эти задачи, но получаемое кусочно-линейное приближение поверхности является слишком грубым. Конечно, в каждой треугольной ячейке сетки можно построить участок гладкой интерполирующей поверхности со сшиванием на границах ячеек [3,5], но такой подход имеет ряд недостатков:

- необходимость задания в узлах сетки не только первых, но и смешанных производных;
- высокая трудоемкость алгоритма гладкой интерполяции;
- сложность расчета сечений на сглаженном участке поверхности.

Для сглаживания поверхности только на ребрах сетки в работе [1] был предложен следующий алгоритм:

1. Найти нормальные вектора касательных плоскостей во всех узлах сетки.
2. Используя методы построения локальных плоских кубических сплайнов [2], вычислить нормали в центральных точках всех ребер сетки.
3. По полученным нормалям (6 на каждый треугольник) построить непрерывную на

Тюкачев Николай Аркадиевич - ВГУ, канд. физ.-мат. наук, доцент, тел.(4732) 208-470, E-mail: nik@cs.vsu.ru

ребрах сетки кусочно-квадратичную вектор-функцию, позволяющую найти нормальные векторы во всех точках области определения.

4. Сглаживать поверхность только на ребрах сетки, используя для этого плоские кубические сплайны. Производные в узлах по требуемым направлениям (наклоны касательных) рассчитывать с помощью векторов нормалей.
5. Вычислять на сглаженных ребрах сетки дополнительные узлы и проводить последовательное сгущение сетки до тех пор, пока это необходимо.

В работе предлагается иной подход, основанный на введении дополнительного разбиения каждого треугольника на несколько новых треугольников, вершины которых лежат на бикубической поверхности Безье.

Для вычисления координат вставляемых точек предлагается следующий алгоритм:

Шаг 1. Разбить каждое ребро триангуляции на nP частей и через эти точки провести в каждом треугольнике линии, порождающие новые треугольники триангуляции.

Шаг 2. В каждой узловой точке вычислить псевдонормаль к поверхности как среднеарифметическое единичных нормалей к треугольникам, сходящимся в этой вершине.

Шаг 3. Вычислить касательные векторы к поверхности вдоль ребер триангуляции, перпендикулярные псевдонормалям.

Шаг 4. На трети расстояния вдоль касательных векторов ввести две промежуточные точки для поверхности Безье.

Шаг 5. Вычислить координаты промежуточных точек по уравнениям, описывающим поверхность Безье.

Для описания линий Безье используется слудующая структура:

```
TLineBezier=record
  p1,p2: word; //№№ точек начала и конца
  A1,A2: TXYZ; //два касательных вектора
end;
TALineBezier=array of TLineBezier;
```

Каждый треугольник порождает три линии Безье и имеет некую серединную точку Pc:

```
  TTrNew=record
    NumLineBezier: array[0..2] of word;
    LineT: array[0..2] of TLineBezier;
    Pc   : TXYZ;    // срединная точка
  end;
```

Элементарной бикубической поверхностью Безье, определяемой матрицей V_{ij}, i=0..3, j=0..3, называется поверхность [6], определяемая векторным параметрическим уравнением

$$r(u,v)=\sum_{i=0}^{3}\sum_{j=0}^{3}C_3^iC_3^ju^i(1-u)^{3-i}v^j(1-v)^{3-j}V_{ij}, \quad (1)$$

где $u,v\in[0,1]$, а число сочетаний из n по i определяется равенством $C_n^i=\frac{n!}{i!(n-i)!}$.

Поверхность Безье обладает следующими свойствами (рис 4):

- является гладкой поверхностью;
- проходит через точки V_{00}, V_{30}, V_{03}, V_{33} и касается отрезков, выходящих из этих точек.

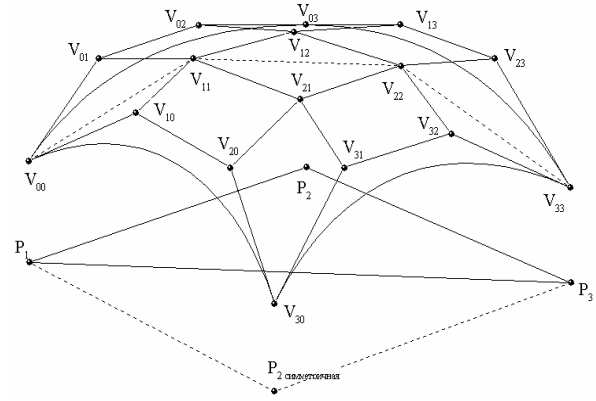

Рис. 4. Поверхность Безье

Предлагаемый алгоритм сглаживания триангуляции состоит из следующих шагов:

Шаг 1. Вычислить единичную нормаль для каждого треугольника.

Шаг 2. Вычислить единичную псевдонормаль в каждой точке триангуляции.

Шаг 3. Сформировать массив LineBezier и массив треугольников TrNew, опирающихся на эти линии.

Шаг 4. Для каждого отрезка массива LineBezier вычислить касательные векторы.

Шаг 5. Для каждого треугольника TrH построить матрицу векторов Безье и новые треугольники.

Шаг 5.1. Построить упорядоченные по обходу линии в треугольнике.

Шаг 5.2. Переставить точки и касательные в LineT.

Шаг 5.3. Вычислить серединную точку $Pc=A_1-P_1+A_2-P_2$

Шаг 5.4. Вычислить матрицу векторов Безье.

Шаг 5.5. Вычислить остальные векторы Безье.

Шаг 5.6. Построить новые треугольники.

Шаг 5.6.1. Вычислить координаты промежуточных точек.

Шаг 5.6.2. Вычислить номера точек треугольника.

Шаг 6. Склеить точки.

Рассмотрим реализацию каждого шага алгоритма более подробно.

Для каждого треугольника нормаль будем вычислять с помощью функции `SetNorm`. Если же проекция нормали на ось z меньше 0, то меняем порядок обхода вершин и направление нормали. Менять порядок обхода вершин необходимо, так как для любого треугольника проекция вектора нормали на ось z должна быть неотрицательна.

Каждая точка триангуляции `PointH[i]` является вершиной нескольких треугольников. Нормированный вектор суммы нормалей к этим треугольникам можно считать некоторым приближением (псевдонормалью) к вектору нормали к поверхности в точке `PointH[i]`.

В дальнейшем потребуется массив отрезков `LineBezier`, каждый элемент которого имеет структуру:

```
TLineBezier=record
  p1,p2: word;
  A1,A2: TXYZ; //два касательных вектора
end;
```

Поля p_1 и p_2 показывают на номера точек начала и конца отрезка, а векторные поля A_1 и A_2 имеют смысл касательных векторов к линии Безье. Вспомогательный массив треугольников опирается не на точки, а на отрезки массива `LineBezier`. Каждый элемент этого массива содержит следующие поля:

```
TTrNew=record
  NumLineBezier : array[0..2] of word;
  LineT    : array[0..2] of TLineBezier;
  Pc       : TXYZ;    // срединная точка
end;
```

В этой структуре поле `NumLineBezier` показывает номер элемента в массиве отрезков `LineBezier`.

На шаге 3 для каждого ребра треугольника p_1,p_2 с помощью функции `FindLine` попытаемся найти элемент в массиве `LineBezier`. Если такого элемента нет, то с помощью функции `AddLine` добавляем его.

Касательные векторы A_1 и A_2 в точках P_1 и P_2 вычисляются по следующему алгоритму (рис. 5):

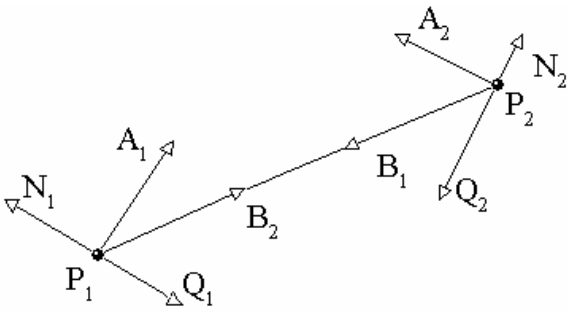

Рис. 5. Касательные векторы A_1 и A_2 в точках P_1 и P_2

Шаг 4.1. Вычисляется вектор отрезка $\bar{A}=\bar{P}_2-\bar{P}_1$;

Шаг 4.2. Вычисляются радиус-векторы точек B_1 и B_2: $\bar{B}_1=\bar{P}_1+\bar{A}/3$, $\bar{B}_2=\bar{P}_2-\bar{A}/3$.

Шаг 4.3. Вычисляются проекции векторов B_1-P_1 и B_2-P_2 на нормали N_1 и N_2.

Шаг 4.4. Находятся касательные векторы A_1 и A_2: $\bar{A}_1=\bar{B}_1-\bar{Q}_1$, $\bar{A}_2=\bar{B}_2-\bar{Q}_2$.

Замечание. Векторы A_1 и A_2 не лежат в одной плоскости.

Шаг 5. Для каждого треугольника `TrNew` необходимо построить матрицу векторов Безье и новые треугольники. Данный этап состоит из следующих шагов.

Шаг 5.1. Построить упорядоченные по обходу линии `LineT` в треугольнике. Вспомогательный массив `LineT: array[0..2] of TLineBezier` содержит для каждого треугольника упорядоченную по обходу информацию о трех отрезках, образующих треугольник: конец каждого отрезка `LineT` является началом для следующего отрезка.

Шаг 5.2. Вычислить (рис. 6) серединную точку $P_c=A_1-P_1+A_2-P_2$ над поверхностью треугольника. Серединная точка P_c потребуется для построения матрицы векторов Безье и должна находиться примерно над серединой треугольника. Праобразы касательных векторов – векторы B_1 и B_2 – равны по длине 1/3 длины своих сторон треугольника. Поэтому сумма векторов B_1+B_2 попадает в центр треугольника P_0 (точку пересечения медиан). Сумма касательных векторов A_1-P_1+A_2-P_2, выходящих из точки P_1, попадет примерно над серединой треугольника.

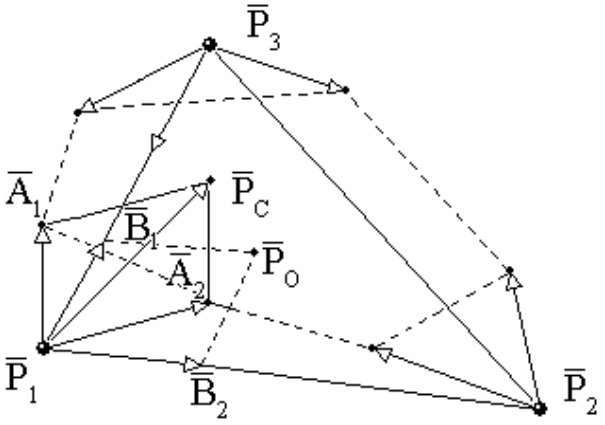

Рис. 6. Серединная точка P_c над поверхностью треугольника

Значение z-координаты для серединной точки берется как среднеарифметическое значение z-координат всех векторов A_1 и A_2, увеличенное в полтора раза. Необходимо отметить, что серединные точки, построенные на основе касательных векторов из точек P2 или P_3, не совпадут с точкой P_c.

Шаг 5.3. Вычислить половину матрицы векторов Безье. Гладкая поверхность Безье, описываемая уравнением (1), определена над четырехугольником. Построим этот четырехугольник, введя точку $P_{2симметричная}$, симметричную точке P_2 относительно середины отрезка P_1P_3. Знание касательных векторов, выходящих из вершин треугольника P_1, P_2 и P_3, и серединной точки P_c позволяет построить 10 из 16 векторов Безье (рис. 4).

Шаг 5.4. Вычислить остальные 6 векторов Безье, считая, что они симметричны соответствующим векторам матрицы Безье относительно середины отрезка $V_{00}V_{33}$ (рис. 4).

Шаг 5.5. Построить новые треугольники, учитывая то, что каждая сторона треугольника разбивается на n_B частей. Это приведет к появлению $nN=(n_B+1)*(n_B+2)/2$ новых точек (рис. 7).

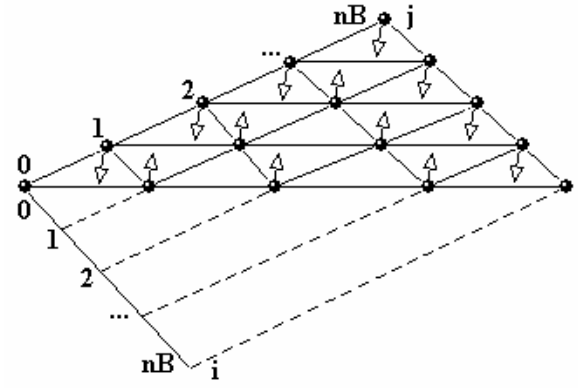

Рис. 7. Новые точки и треугольники

Далее необходимо вычислить координаты новых точек и занести их в массив `PointTemp`.

Шаг 5.5.1. Вычислить координаты промежуточных точек с помощью уравнения (1). Все новые точки будем собирать в массиве `PointTemp: array of TXYZ`. Поэтому для следующего шага построения новых треугольников, которые опираются на точки массива `PointTemp`, потребуются две функции:

функция `IJ_in_N`, которая по номеру строки точек i_0 и номеру точки в строке j_0 возвращает абсолютный номер точки в массиве `PointTemp`;

- обратная процедура `N_in_IJ`, которая по абсолютному номеру точки вычисляет номер строки i и номер точки в строке j.

Шаг 5.5.2. Вычислить номера точек треугольника и добавить новые треугольники. Из рис. 7 видно, что точки с номерами (0,0) и (n_B,n_B) не добавляют новых треугольников. Часть точек добавляет по одному треугольнику, остальные добавляют по два новых треугольника.

Шаг 6. Склеить точки. Построенный массив точек `PointTemp` является вспомогательным и избыточным: может оказаться, что для двух соседних треугольников начальной триангуляции соседние точки на границе близки. Потому при построении основного массива точек `PointH` необходимо добавлять только существенно отличающиеся точки, отстоящие друг от друга на расстояние больше `Eps`.

Этот шаг очень важен, так как множество добавленных точек для каждого треугольника свое,а соответствующие добавленные точки на границе двух соседних треугольников могут не совпадать (рис. 8).

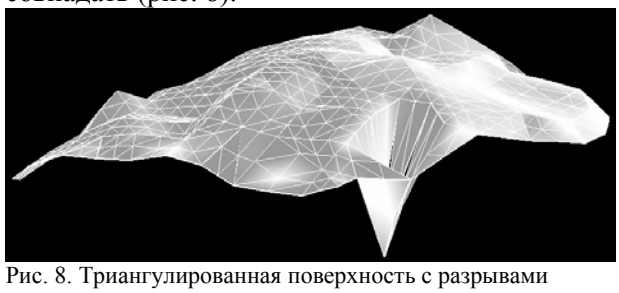

Рис. 8. Триангулированная поверхность с разрывами

На рис. 8 видны черные линии между несклеенными треугольниками. Фактически данная операция склеивает поверхность треугольников в непрерывную.

Вместо проверки близости соответствующих добавленных точек на границе двух соседних треугольников можно использовать топологический алгоритм: если у соседнего

треугольника уже вычислены добавленные точки, то не вычислять граничные точки, а использовать готовые.

Такам образом, в работе предлагается алгоритм сглаживания триангулированной поверхности, основанный на введении дополнительного разбиения каждого треугольника на несколько новых треугольников, вершины которых лежат на бикубической поверхности Безье. Алгоритм использован при разработке геоинформационной системы «Kemberlit Explorer».

Литература

1. Костюк Ю.Л. Визуально гладкая аппроксимация однозначной поверхности, заданной нерегулярным набором точек // Ю.Л. Костюк, А.Л. Фукс. Теория геоинформатики и дистанционного зондирования. – Томск: Изд-во Том. ун-та, 2003, с. 41-45.
2. Костюк Ю.Л. Применение сплайнов для изображения линий в машинной графике // Автоматизация эксперимента и машинная графика. – Томск: Изд-во Том. ун-та, 1977, с. 116-130.
3. Роджерс Д., Адамс Дж. Математические основы машинной графики. – М.: Мир, 1980. – 240 с.
4. Тюкачев Н.А. Сглаживание триангуляции трехмерных поверхностей // Информатика: проблемы, методология, технологии: мат. 6 межд. науч.-мет. конф., Воронеж: ВГУ, 2006. с. 9-11.
5. Шенен П., Коснар М., Гардан И. и др. Математика и САПР: В 2 кн. Кн. 1 / Пер. с франц. – М.: Мир, 1988. – 204 с.
6. Шикин Е.В. Компьютерная графика. Полигональные модели. / Е.В. Шикин, А.В. Боресков. – М.: ДИАЛОГ-МИФИ, 2005. – 223 с.

Воронежский государственный университет

SMOOTHING OF THE TRIANGULATED SURFACE

N.A. Tjukachev

The algorithm of smoothing triangulated surface, based on introduction of additional splitting of each triangle on several new triangles with vertexes on bicubic Bezier surface

Keywords: geometrical algorithms, smoothing, Bezier