Федеральное агентство по образованию РФ
ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
Факультет информатики
Кафедра теоретических основ информатики

УДК 681.03

ДОПУСТИТЬ К ЗАЩИТЕ В ГАК
Зав. кафедрой, д.т.н.
_____ Ю.Л. Костюк
«__» _____ 2005 г.

Юсупов Александр Рашитович

РАСЧЁТ ВЫСОТ СТРУКТУРНЫХ ЛИНИЙ ДЛЯ ПОСТРОЕНИЯ ЦИФРОВОЙ МОДЕЛИ РЕЛЬЕФА

Дипломная работа

Научный руководитель,
к.т.н., доцент _____ А.Л. Фукс

Исполнитель,
студент гр. 1401 _____ А.Р. Юсупов

Электронная версия диплома работы помещена
в электронную библиотеку. Файл
Администратор

Томск – 2005

Реферат

Дипломная работа 24 с., 11 рис., 7 источников

ЦИФРОВАЯ МОДЕЛЬ РЕЛЬЕФА, ПРЕДОБРАБОТКА СТРУКТУРНЫХ ЛИНИЙ РЕЛЬЕФА, ПРОВЕРКА ВЫСОТ СТРУКТУРНЫХ ЛИНИЙ, ТРИАНГУЛЯЦИЯ ДЕЛОНЕ С ОГРАНИЧЕНИЯМИ, , BORLAND C++ BUILDER 6.0.

Объект исследования – методы предварительной обработки структурных линий и высотных отметок рельефа для построения высококачественной цифровой модели рельефа.

Цель работы – разработка и реализация алгоритмов проверки и интерполяции значений высот линий и точек в рамках программной системы предобработки исходных данных для построения цифровой модели рельефа.

Методы исследования – изучение алгоритмов триангуляции с сильными ограничениями, разработка собственных алгоритмов и необходимых структур данных, использование среды разработки программного обеспечения Borland C++ Builder 6.0.

Разработаны и реализованы алгоритмы интерполяции высот в узлах трехмерных структурных линий, проверки перепадов высот, триангуляции Делоне с сильными ограничениями.

Реализованные алгоритмы включены в систему предварительной обработки данных для построения цифровых моделей рельефа. В рамках данной системы созданы модули для работы с проектом – описанием наборов исходных данных и легенды для их графического представления.

Содержание

Введение

Важным объектом исследования геоинформационных систем (ГИС) является земной рельеф. Как правило, рельеф задается наборами высотных отметок и структурных линий, которые получаются с помощью векторизации карт или методов дистанционного зондирования. Структурные линии определяют множества точек с одинаковыми значениями высоты (основные и промежуточные изолинии) или резким изменением наклона рельефа (береговые линии рек и озер, границы оврагов, обрывов, промышленных выработок и т.д.). Линии последнего типа называют трехмерными или 3D-линиями, т.к. в отличие от изолиний они не лежат в одной горизонтальной плоскости. Все структурные линии задаются наборами узловых точек и считаются ломаными.

Для работы с рельефом необходимо построить его цифровую модель (ЦМР) на основе прямоугольной или треугольной сетки, в узлах которой заданы высоты. Высотные отметки и структурные линии желательно сохранить в получаемой модели рельефа, поэтому ЦМР обычно строят на основе треугольной сетки. При этом расчет ЦМР производится в два этапа [1].

На первом этапе по всем высотным отметкам и вершинам линий строится триангуляция Делоне. Каждая структурная линия задает некоторое ограничение на форму поверхности рельефа, поэтому на втором этапе производится включение отрезков линий в ЦМР. В результате получается триангуляция со слабыми ограничениями, в которой каждый отрезок совпадает с некоторым ребром триангуляции.

В каждой высотной отметке и каждом узле линии должна быть определена высота. Как правило, в наборах исходных данных для линии задается только одно значение высоты. В случае изолиний этого вполне достаточно, но для 3D-линий значение высоты приходится просто считать пропущенным, и в узлах линии необходим дополнительный расчет высот. Если некоторые вершины 3D-линий совпадают с высотными отметками или через них проходят изолинии, то такой расчет можно провести на основе интерполяции. Однако во многих случаях только исходных точек и линий для интерполяции недостаточно, поэтому требуется расчет дополнительных отметок с автоматическим или интерактивным определением их высот.

Триангуляция со слабыми ограничениями, в которой для каждой вершины задано значение высоты, определяет кусочно-линейную однозначную поверхность. Эта поверхность используется в большинстве известных ГИС как цифровая модель рельефа. Однако качество данной модели невысоко, т.к. она всегда содержит недопустимые плоские горизонтальные участки. В принципе, горизонтальные участки рельефа могут существовать (например, это может быть промышленная площадка или поверхность озера), но границей такого участка должна быть 3D-линия. Недопустимые горизонтальные участки появляются там, где ЦМР содержит треугольники, все вершины которых принадлежат одной изолинии. В том, что это является нарушением, легко убедиться, если на основе ЦМР получить расчетные изолинии – отличия их от исходных будут очень существенными. Для устранения данного нарушения необходимо построить триангуляцию с сильными ограничениями: рассчитать дополнительные высотные отметки, добавить их в триангуляцию и перестроить все недопустимые участки.

Решению задач проверки и восстановления высот в узлах 3D-линий и построения триангуляции с сильными ограничениями посвящена представляемая дипломная работа.

1. Проверка и восстановление пропущенных значений высот

Во многих геоинформационных системах важным объектом исследования является рельеф местности. Как правило, рельеф задается наборами высотных отметок и структурных линий [1, 4]. Высотные отметки обычно представляют локальные минимумы или максимумы, а также другие характерные точки рельефа. Структурные линии накладывают дополнительные ограничения на форму поверхности рельефа. Для получения исходных линий и точек используются методы дистанционного зондирования, полевые исследования и векторизация существующих картографических материалов. В последнем случае структура исходных данных бывает наиболее сложной, поэтому будем ориентироваться именно на него.

Существует два основных типа структурных линий. Во-первых, это изолинии (горизонтали), которые определяют множества точек с одинаковыми значениями высоты. В картографии различают основные изолинии и промежуточные, которые используются для более точного задания сложных участков рельефа. Основные изолинии всегда либо замкнуты, либо выходят на границу карты. Промежуточные горизонтали могут быть любыми.

Второй тип линий – трехмерные или 3D-линии. Эти линии определяют множества точек, в которых наклон рельефа изменяется скачком. К 3D-линиям относятся границы оврагов, обрывов, береговые линии, карьерные выработки и т.д.

Все линии задаются наборами узловых точек и для упрощения их дальнейшей обработки считаются ломаными. На рис. 1 изображен векторизованный планшет карты (3D-линии более светлые).

Для работы с рельефом необходимо построить его цифровую модель (ЦМР) на основе прямоугольной или треугольной сетки, в узлах которой заданы высоты. Высотные отметки и структурные линии желательно сохранить в получаемой модели рельефа, поэтому ЦМР обычно строят на основе треугольной сетки. При этом расчет ЦМР производится в два этапа [1].

На первом этапе по всем высотным отметкам и вершинам линий строится триангуляция Делоне. Каждая структурная линия задает некоторое ограничение на форму поверхности рельефа, поэтому на втором этапе производится включение отрезков линий в ЦМР. В результате получается триангуляция со слабыми ограничениями, в которой каждый отрезок совпадает с некоторым ребром триангуляции.

Качество получаемой модели рельефа определяется, прежде всего, качеством исходных данных. Поэтому важным этапом является предварительная проверка и корректировка линий и отметок. Типичными ошибками исходных данных являются:

1. Разрывы линий на границах смежных планшетов карт, связанные с неточностями самих карт и/или их координатной привязки при векторизации. Данное нарушение приводит к скачкообразному изменению высот на границах планшетов.
2. Вырожденные отрезки (расстояние между соседними вершинами линии меньше заданного порогового значения).
3. Самопересечение линий.
4. Полное или частичное совпадение линий. Эта ошибка может появиться во время векторизации и не обнаруживается визуально.
5. Примыкание или пересечение изолиний разных уровней, попадание отметки одной высоты на изолинию другой высоты. Это недопустимо, т.к. поверхность рельефа должна быть однозначной.

6. Наличие висячих вершин изолиний. Если основная изолиния незамкнута, то ее конечные точки должны лежать либо на границе планшета, либо на некоторой трехмерной линии. Промежуточная изолиния может быть любой.

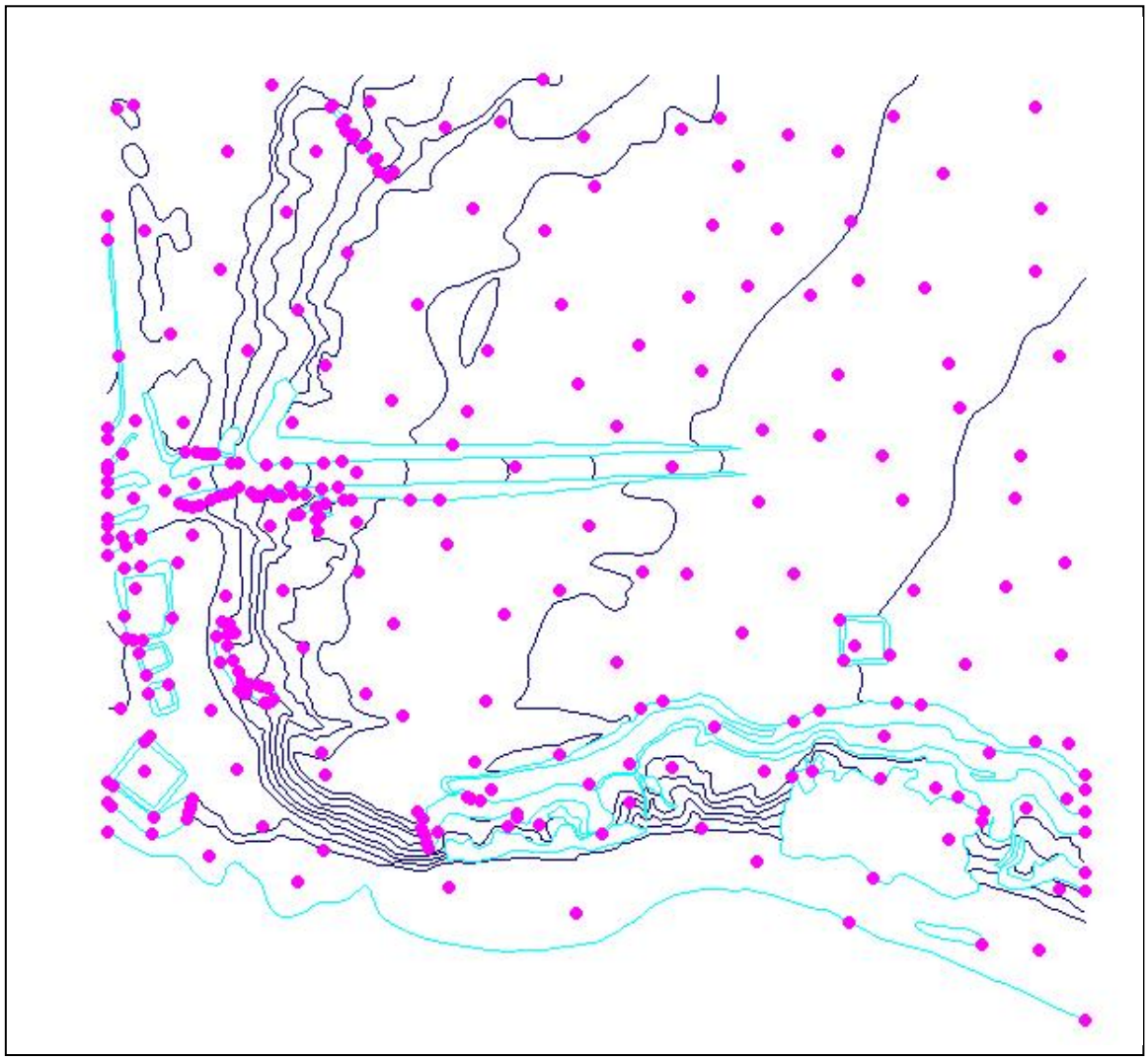


Рис. 1. Исходные данные для построения цифровой модели рельефа

Ошибки, связанные с нарушением однозначности поверхности рельефа, вообще не позволяют построить корректную ЦМР. Они должны быть устранены в первую очередь. Но существуют и другие нарушения, без исправления которых триангуляция с ограничениями еще не дает нужного результата. Это, прежде всего, ошибки в задании высот. Их практически невозможно обнаружить визуально, поэтому проверка высот всегда проводится в автоматическом режиме. Что же касается корректировки ошибок, то ее чаще всего приходится проводить интерактивно.

Отдельная проблема возникает при использовании 3D-линий. Для получения корректной ЦМР во всех высотных отметках и узлах линий должны быть определены высоты. В случае изолинии достаточно задать одно общее значение для всех вершин. Но измерить высоты в узлах 3D-линий, как правило, не удается, и они остаются пропущенными. Поэтому в узлах трехмерных линий необходим дополнительный расчет (восстановление) высот.

Пусть в двух произвольных узлах 3D-линии *A* и *B* высоты уже известны. Тогда для расчета высот во всех вершинах между *A* и *B* можно использовать интерполяцию (проще всего, линейную). Для полного восстановления высот необходимо разбить все трехмерные линии на участки (дуги), в конечных точках которых высоты уже вычислены. Рис. 2 показывает, какие точки можно непосредственно использовать в качестве концевых точек дуг 3D-линий (более светлых линий):

- точки, в которых к 3D-линии примыкают изолинии;
- точки 3D-линий, совпадающие с высотными отметками.

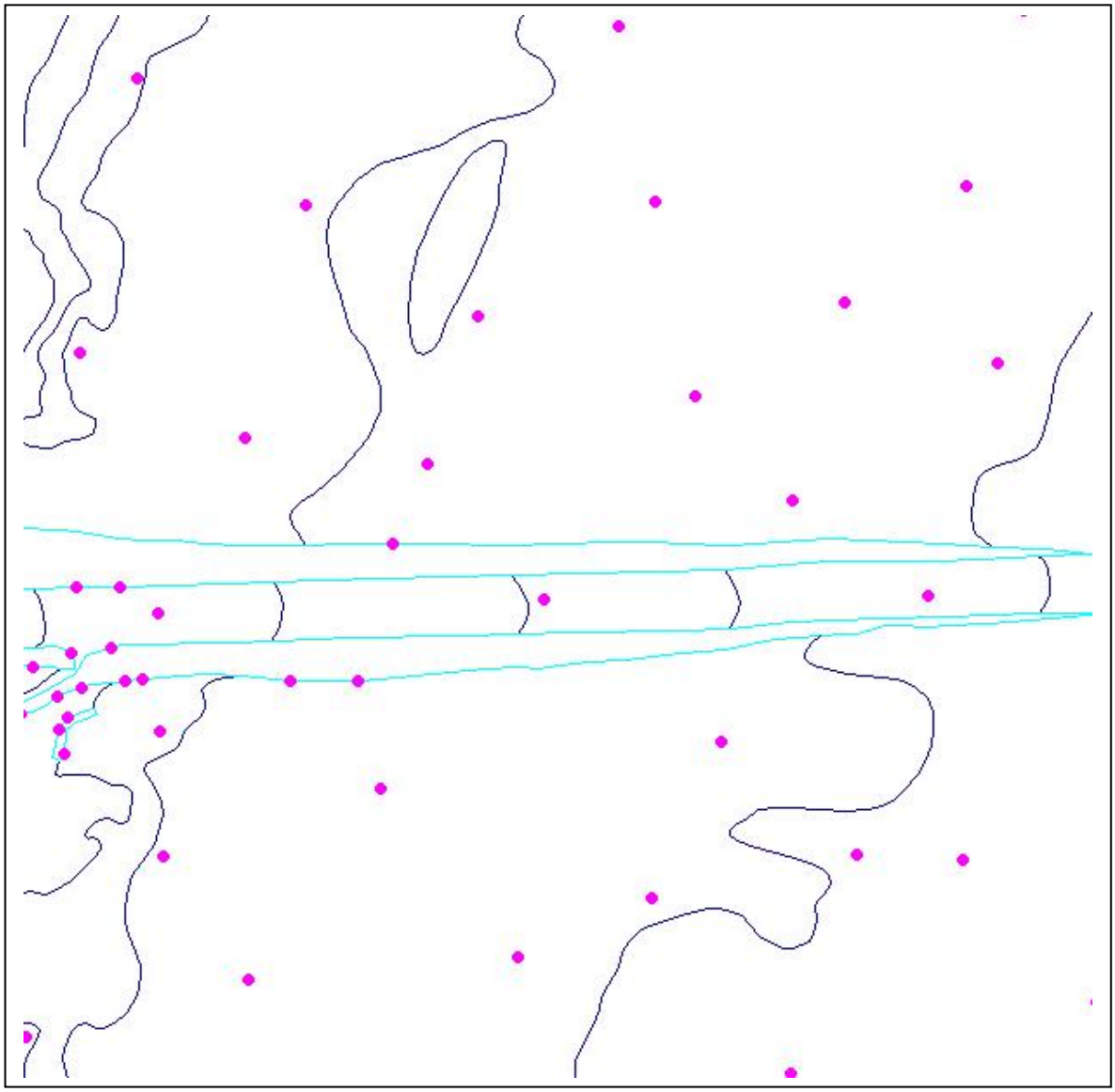


Рис. 2. 3D-линия: высотные отметки в вершинах и примыкающие изолинии

Очень важно, что высота в этих точках уже известна – это высота соответствующей изолинии или отметки. К сожалению, только этих легко выделяемых точек недостаточно. В некоторых случаях придется автоматически вычислять концевые точки дуг и устанавливать в них дополнительные отметки, высоты которых можно определить только в интерактивном режиме. Это необходимо сделать:

- в висячих конечных вершинах незамкнутых трехмерных линий;

- в точках пересечения двух и более 3D-линий.

Отдельно выделим случай замкнутой линии. Для восстановления высот в ее узлах, в принципе, достаточно определить высоту только в ее начальной (конечной) точке. Но тогда трехмерная линия станет, фактически, изолинией. Обычно так и должно быть (например, для береговой линии озера), но если линия действительно является трехмерной, устанавливать дополнительные отметки и определять их высоту придется в интерактивном режиме.

Алгоритм выделения концов дуг 3D-линий основан на простых проверках совпадения точек, пересечения линий и попадания точек на линии. Однако общее число проверок составляет $O(n^2)$ для n вершин линий и высотных отметок. Значение n часто достигает десятков тысяч, поэтому алгоритм с квадратичной трудоемкостью будет малопригодным.

Предложим более быстрый алгоритм. Пусть по набору исходных структурных линий и высотных отметок построена триангуляции со слабыми ограничениями (в большинстве случаев это можно сделать за $O(n)$). В результате выполнятся следующие условия:

- если две или более линий пересекаются, то точка пересечения является вершиной каждой из них;
- если высотная отметка находится на линии, то она обязательно совпадает с вершиной этой линии;
- если некоторая линия примыкает к другой, то точка примыкания обязательно является вершиной обеих линий;
- если конечная вершина линии висячая, то она не совпадает с какой-либо другой точкой.

Пусть в процессе триангуляции строились также списки точек, совпадающих с вершинами треугольников (для каждой точки, добавляемой к триангуляции, обязательно проверяется, не совпадает ли она с какой-нибудь вершиной). Последовательно просматривая все вершины триангуляции и списки совпадающих с ними точек, можно легко выделить все концы дуг трехмерных линий. Проверка концов дуг и совпадающих с ними точек позволяет определить, известно ли уже значение высоты, или необходимо автоматически добавить новую высотную отметку и интерактивно задать ее высоту. Трудоемкость такого алгоритма восстановления высот будет линейной.

Рис. 3 иллюстрирует результат работы алгоритма восстановления высот на основе триангуляции с ограничениями. Черными точками отмечены концы выделенных дуг трехмерных линий.

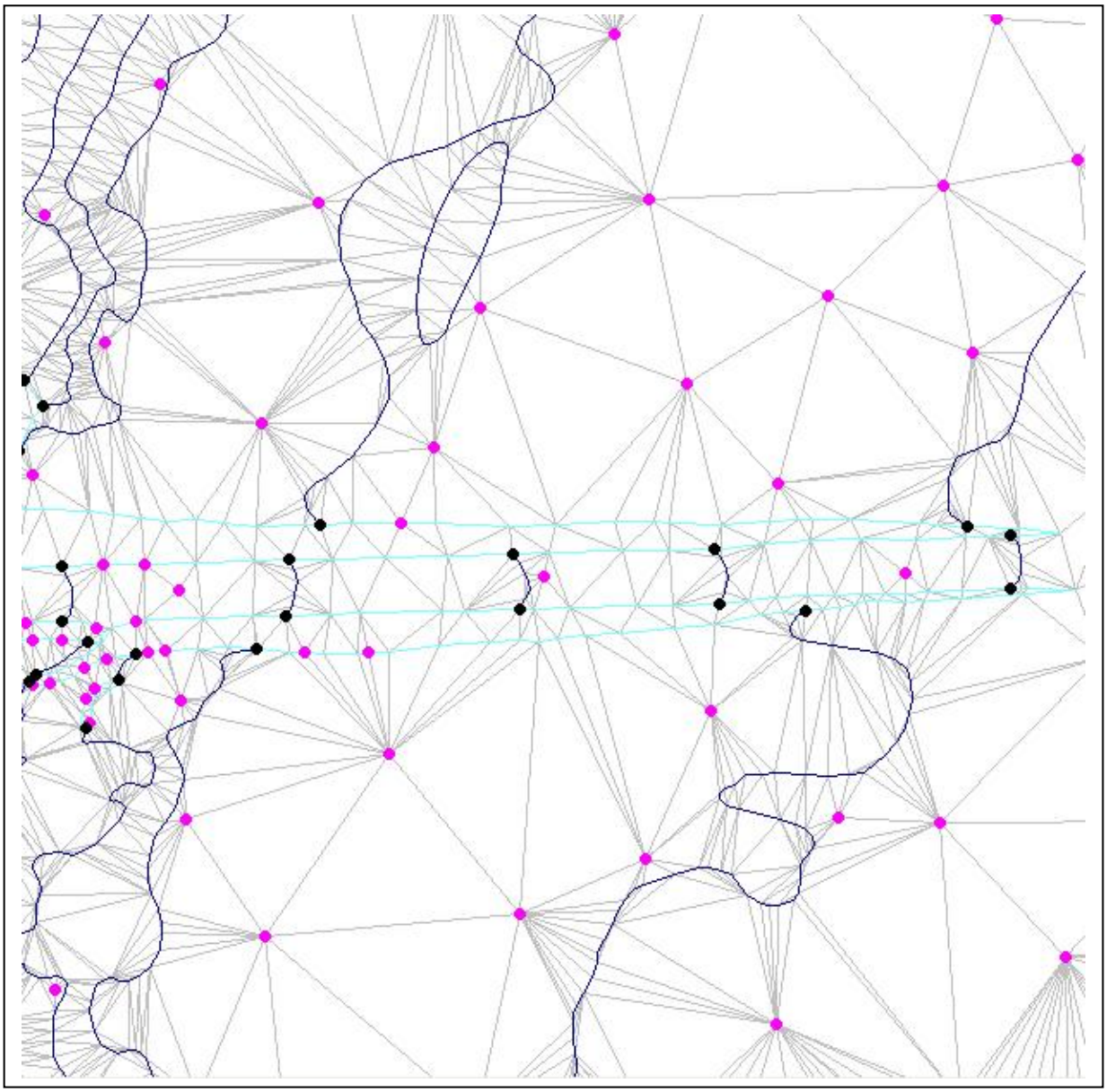


Рис. 3. Выделение дуг 3D-линий на основе триангуляции со слабыми ограничениями

2. Построение цифровой модели рельефа на основе триангуляции с сильными ограничениями

В большинстве геоинформационных систем построение цифровой модели рельефа проводится в два этапа. Вначале по всем высотным отметкам и узловым точкам структурных линий строится триангуляция Делоне. Затем в полученную систему треугольников включаются все отрезки структурных линий таким образом, что каждый отрезок будет совпадать с ребром некоторого треугольника [1, 4]. Полученная система треугольников называется триангуляцией со слабыми ограничениями. Если во всех вершинах треугольников задана высота, то триангуляция определяет кусочно-линейную однозначную поверхность, которая и используется в качестве цифровой модели рельефа.

Но учет только слабых ограничений оказывается недостаточным. Качество данной модели невысоко, т.к. она всегда содержит недопустимые плоские горизонтальные участки. В принципе, горизонтальные участки рельефа могут существовать (например, это может быть промышленная площадка или поверхность озера), но границей такого участка должна быть 3D-линия. Недопустимые горизонтальные участки появляются там, где ЦМР содержит треугольники, все вершины которых принадлежат одной изолинии или изолиниям одного уровня. В том, что это является нарушением, легко убедиться, если на основе ЦМР получить расчетные изолинии – отличия их от исходных будут очень существенными.

Для устранения данного нарушения необходимо ввести новый тип ограничений – сильные. Будем считать, что пространственная триангуляция удовлетворяет сильным ограничениям, если для нее выполняются слабые ограничения, и, кроме того, в ней отсутствуют горизонтальные ребра, лежащие на уровнях изолиний и не совпадающие с отрезками структурных линий. Такие ограничения гарантируют отсутствие плоских горизонтальных участков, по крайней мере, на заданных уровнях.

Назовем недопустимыми все ребра, лежащие на уровне изолиний, но не совпадающие с отрезками изолиний (горизонтальные ребра). Любой треугольник, для которого не выполняются сильные ограничения, может содержать одно, два или три недопустимых ребра (рис. 4). Причинами образования таких треугольников являются сложные извилистые формы изолиний, наличие седловых точек поверхности и замкнутых линий, внутри которых нет высотных отметок. Примеры участков триангуляции, на которых нарушаются сильные ограничения, приведены на рис. 5.

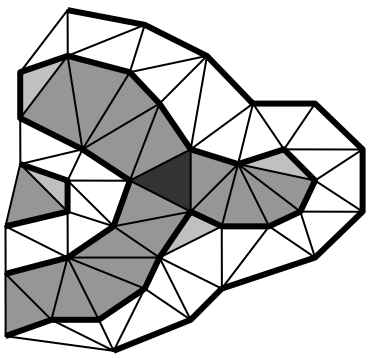
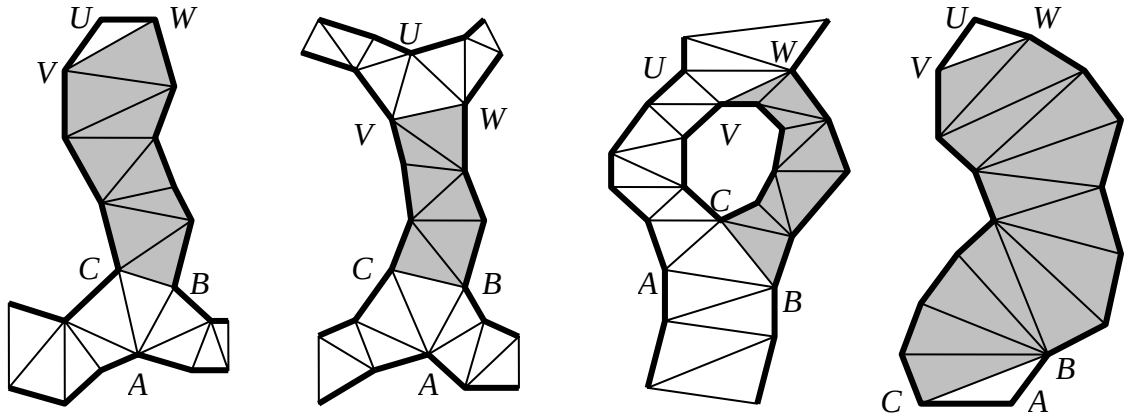


Рис. 4. Треугольники с одним (светлая закраска), двумя (темная закраска) и тремя (черная закраска) недопустимыми ребрами

При обработке реальных данных число недопустимых горизонтальных треугольников в триангуляции со слабыми ограничениями может быть весьма велико. Это иллюстрирует рис. 6, на котором недопустимые треугольники заштрихованы. Поэтому проверка сильных ограничений и соответствующее перестроение триангуляции должны проводиться в автоматическом режиме.



(а) (б) (в) (г)

Рис. 5. Примеры треугольников, содержащих недопустимые ребра (выделены серым цветом): (а) – между изолиниями разных уровней, (б) и (в) – на участках с седловой точкой, (г) – внутри замкнутой изолинии

Для получения триангуляции с сильными ограничениями необходимо рассчитывать дополнительные высотные отметки и добавлять их в триангуляцию до тех пор, пока не будут перестроены все недопустимые треугольники. Прежде всего, нужно избавиться от треугольников, содержащих по три недопустимых ребра.

Горизонтальный треугольник *ABC* с тремя недопустимыми ребрами на уровне z_A интерполирует участок рельефа, который реально также является почти горизонтальным. Будем считать, что центр *ABC* (точка *O*) отклоняется от z_A по высоте на фиксированную величину Δz. Однако необходимо правильно определить знак такого отклонения. Для этого достаточно проверить треугольники в окрестности *ABC*, переходя по горизонтальным ребрам к соседним треугольникам до тех пор, пока не будет достигнут любой негоризонтальный треугольник *UVW*. Пусть, например, $z_U \neq z_A$, тогда знак Δz определяется знаком отклонения $z_A - z_U$, а также количеством отрезков изолиний, пройденных на пути от *ABC* до *UVW* (при переходе через отрезок изолинии знак меняется на противоположный).

Задав в *O* высоту $z_A \pm \Delta z$, можно заменить *ABC* на три треугольника *ABO*, *BCO*, *CAO*, содержащих по одному недопустимому ребру.

Если перестроить все треугольники с тремя недопустимыми сторонами, то в триангуляции останутся только цепочки треугольников, содержащие одно или два недопустимых ребра. Все эти цепочки выделяются очень легко, т.к. они начинаются и заканчиваются треугольниками, содержащими одно недопустимое ребро. Примеры таких цепочек были приведены на рис. 5 (заштрихованы треугольники с двумя недопустимыми сторонами).

Рис. 6. Участки с нарушениями сильных ограничений триангуляции

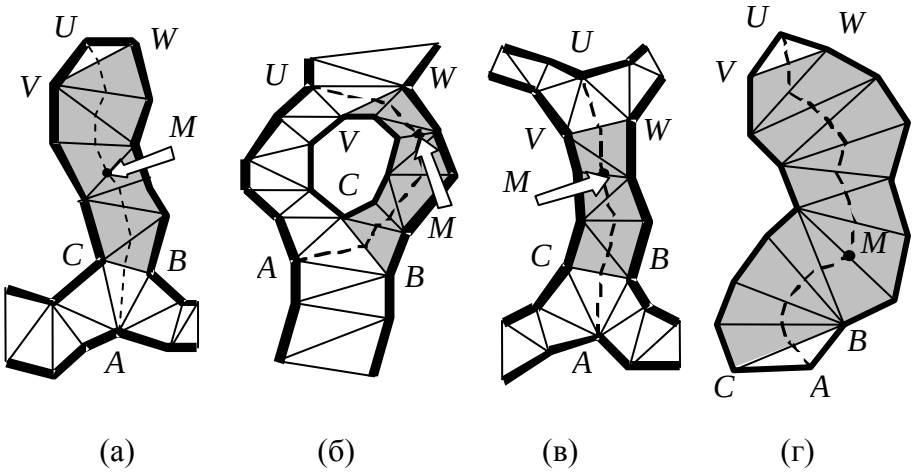
В алгоритме перестроения цепочек горизонтальных треугольников анализируются различные варианты взаимного расположения треугольников и изолиний, и в соответствии с ними производится добавление новой высотной отметки и перестроение триангуляции. Это иллюстрирует рис. 7.

Пусть некоторая цепочка начинается с треугольника *ABC*, а заканчивается треугольником *UVW*, причем *BC* и *VW* – недопустимые ребра, т.е. $z_B = z_C = z_V = z_W$, а точки *B*, *C*, *V* и *W* являются вершинами изолиний одного уровня. Предположим, что ломаная, проходящая из *A* в *U* через центры недопустимых ребер, имеет длину *L*, а длина самого короткого недопустимого ребра цепочки равна *l*. Вычислим на ломаной новую точку *M* по следующим правилам:

1. Если $z_A \neq z_U$, то полагаем, что высота вдоль ломаной от *A* до *U* изменяется линейно. Тогда *M* – это равноудаленная от *A* и *U* точка ломаной (рис. 7а), и $z_M = (z_A + z_U)/2$.

2. Если $z_A = z_U \neq z_B$, то поверхность имеет седловую точку. Будем считать, что седловой точкой M является точка пересечения ломаной и самого короткого недопустимого ребра цепочки (рис. 7б и 7в), причем $z_M = \dfrac{z_B \cdot L + z_A \cdot l}{L + l}$.

3. Если $z_A = z_U = z_B$, то цепочку образуют треугольники внутри замкнутой изолинии. Естественно предположить, что соответствующий участок рельефа также является почти горизонтальным. Будем считать, что наибольшее отклонение от горизонтальной плоскости $z = z_B$ достигается в точке M – центре самого длинного недопустимого ребра (рис. 7г), а высота $z_M = z_B \pm \Delta z$, где знак отклонения определяется так же, как и для треугольников с тремя недопустимыми ребрами.



(а) (б) (в) (г)

Рис. 7. Проверка цепочек треугольников с недопустимыми ребрами и добавление новой точки М: (а) – между изолиниями разных уровней, (б) и (в) – на участках с седловой точкой, (г) – внутри замкнутой изолинии

После этого точка М включается в триангуляцию Делоне. При перестроении треугольников проверяется, не пересекает ли добавляемое ребро триангуляции какой-нибудь отрезок структурной линии. Если такое пересечение есть, то точка пересечения добавляется и структурную линию, и в триангуляцию. Таким образом, полученная система треугольников будет оставаться триангуляцией Делоне со слабыми ограничениями.

После добавления точки М цепочка разбивается на две более короткие (возможно, уже пустые), а общее число недопустимых ребер уменьшается. Процесс заканчивается, когда недопустимых ребер в триангуляции не останется. Такой алгоритм позволяет перестроить триангуляцию с учетом сильных ограничений, добавляя минимальное количество точек.

Рис. 8 иллюстрирует процесс проверки сильных ограничений: пока не перестроенные треугольники с недопустимыми ребрами заштрихованы, добавленные высотные отметки изображены черными точками.

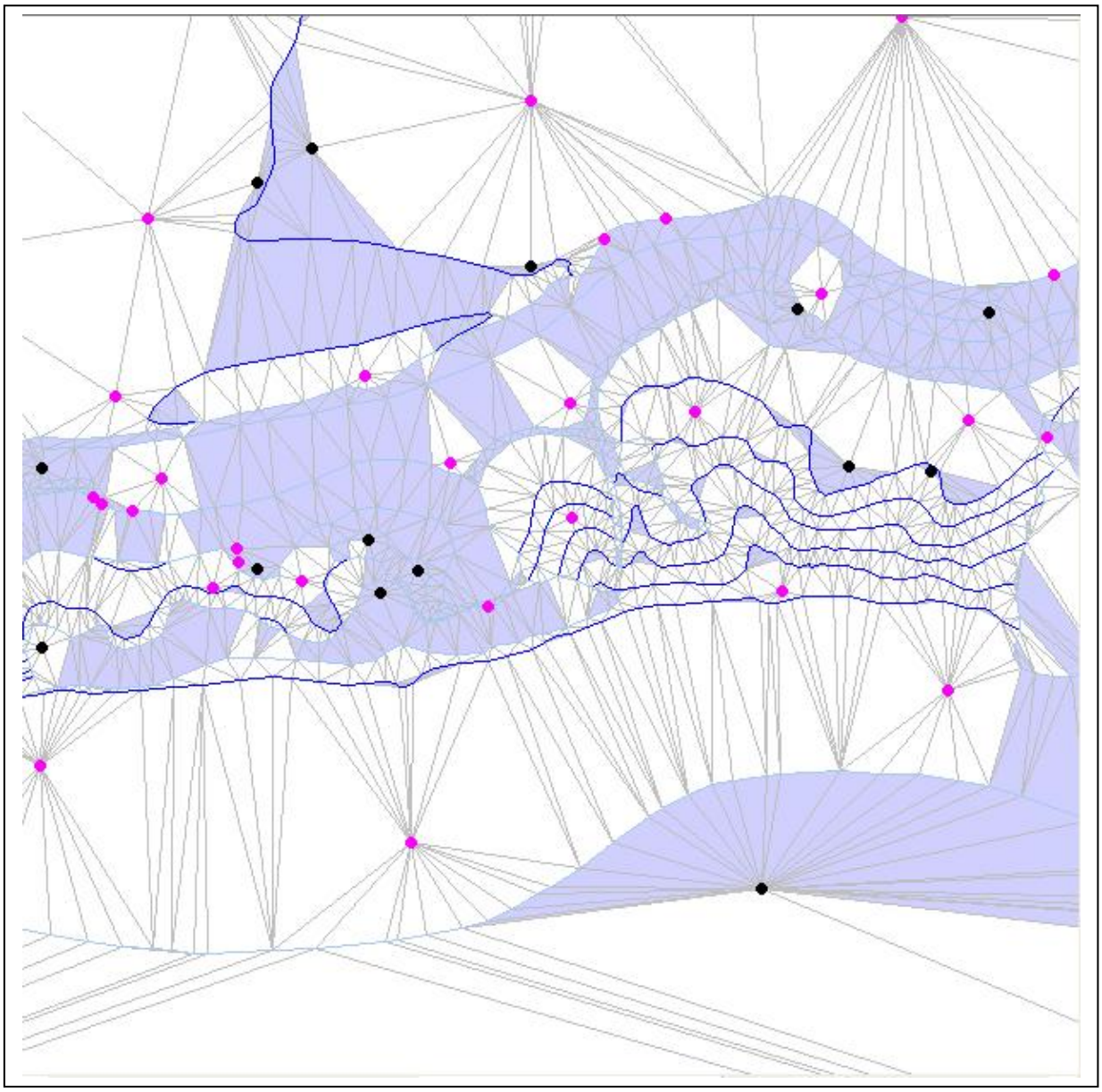


Рис. 8. Проверка сильных ограничений и добавление новых высотных отметок

3. Руководство пользователя

Дипломная работа посвящена созданию системы предварительной обработки данных для построения цифровой модели рельефа. Программной средой для разработки являлась система Borland C++ Builder 6.0 [5].

Прежде всего, нужно было разработать систему классов, представляющих объекты разных типов. Элементарные объекты описывают классы CPoint (высотные отметки и вершины линий), CLine (структурные линии рельефа) и CTrian (треугольники ЦМР). Класс CDEMData объединяет наборы линий, точек, треугольников, а также основные методы, связанные с проверкой данных и построением ЦМР. Исходные данные хранятся в обменном формате шейп-файлов системы ArcView [6], описатели шейп-файлов представляет класс CShape.

Для управления исходными и выходными данными в рамках настоящей дипломной работы был создан класс CProject. Этот класс представляет набор используемых шейп-файлов и легенду – цвета и типы линий, применяемые для отображения структурных линий и высотных отметок.

Класс CProject позволяет создать новый проект или загрузить существующий, а также сохранить текущий набор шейпов и легенду. Все действия, выбираемые пользователем, выполняются с помощью соответствующих методов класса CProject. Пользователь системы не имеет прямого доступа к структурам данных, представляющим ЦМР, и не может их повредить.

На рис. 10 приводится главная форма программы, которая содержит окно проекта с именами и легендой используемых шейпов, а также основное окно, в котором отображаются структурные линии и высотные отметки. В верхней части формы находиться главное меню, с помощью которого пользователь может управлять всеми действиями программы. В нижнем левом углу отмечается текущее состояние или выполняемое действие.

Перейдём к рассмотрению пунктов меню. Начнём с меню **Файл** (рис. 9).

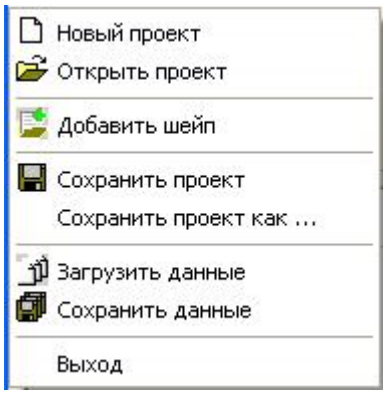


Рис. 9. Меню Файл для работы с шейп-файлами и проектом

Первый пункт меню **Файл** – это создание нового проекта. При начальной загрузке программы новый проект создаётся автоматически. Пользователь может загрузить другой проект или добавить новые шейпы в уже существующий.

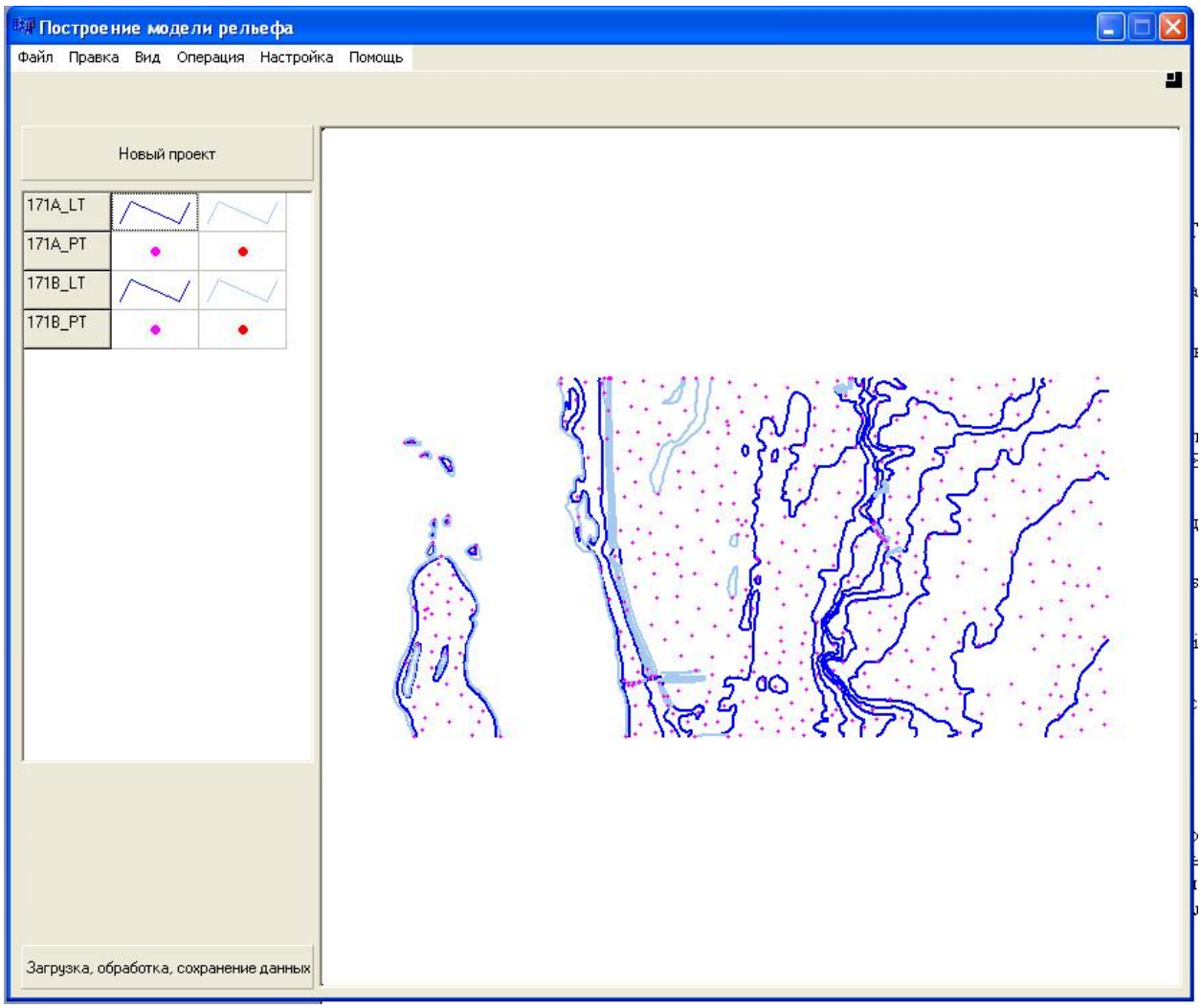


Рис. 10. Общий вид главной формы системы предобработки данных

Второй пункт меню **Файл** – это открытие ранее сохранённого проекта. Если при выборе данного пункта уже есть открытый проект, то пользователю будет предложено сохранить его. Далее пользователь должен выбрать файл проекта, который он хочет загрузить.

Третий пункт меню **Файл** – добавление нового шейпа в проект. При выборе этого пункта пользователю будет предложено выбрать шейп-файл, который он хочет загрузить.

Четвёртый пункт – сохранение проекта. Если проект до этого не сохранялся, то пользователю будет предложено указать расположение сохраняемого проекта и его имя. В противном случае проект будет сохранён под старым именем.

Пятый пункт меню **Файл** – сохранение проекта с новым именем. При выборе этого пункта текущим проектом становится тот, который был указан в диалоге сохранения.

Шестой пункт меню **Файл** – загрузка данных. При его выборе из всех шейпов текущего проекта исходные данные загружаются в соответствующие наборы и отображаются в основном окне главной формы (рис 10).

Седьмой пункт меню **Файл** – сохранение данных в соответствующих шейп-файлах проекта.

Меню **Правка** позволяет выполнять все действия, связанные с интерактивным изменением точек и линий.

Команды меню **Вид** позволяют увеличивать и уменьшать масштаб изображения, сохранять и выбирать окна для изображения, работать с набором обнаруженных ошибок, а также выбирать, какие объекты будут изображаться в основном окне формы.

Меню **Операция** (рис. 11) представляет основные операции предобработки структурных линий и высотных отметок.

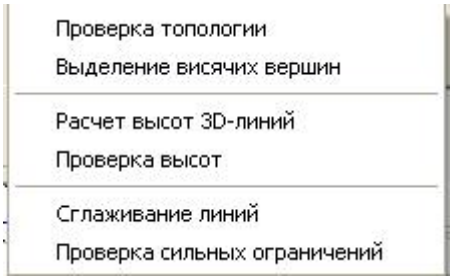


Рис. 11. Пункты меню Операция

При выборе пункта **Расчёт высот 3-D линий** начинает работу алгоритм восстановления значений высот в вершинах трехмерных линий. Прежде всего, будет построена триангуляция со слабыми ограничениями (если она еще не получена). Далее будут автоматически выделяться концы дуг всех 3D-линий. Если в конечной точке дуги не удается определить значение высоты, то в этом месте устанавливается новая высотная отметка, и пользователь должен задать ее высоту сам.

При выборе пункта **Проверка сильных ограничений** вначале строится триангуляция со слабыми ограничениями, а затем выполняется алгоритм проверки сильных ограничений. Новые отметки и их высоты рассчитываются автоматически. Результирующая система треугольников будет триангуляцией Делоне, для которой выполняются и слабые, и сильные ограничения.

4. Руководство программиста

Дипломная работа посвящена созданию системы предварительной обработки данных для построения цифровой модели рельефа. Программной средой для разработки являлась система Borland C++ Builder 6.0 [5].

Класс CDEMData объединяет основные наборы, представляющие ЦМР - точек, линий, треугольников - и параметры области определения.

```
class CDEMData
{
protected:

        CSet<CPoint> m_Points;                  // набор точек
        CSet<CLine> m_Lines;                    // набор линий
        CSet<CTrian> m_Trians;                  // набор треугольников

        double m_Xmin, m_Xmax;                  // диапазон значений X
        double m_Ymin, m_Ymax;                  // диапазон значений Y
        double m_Zmin, m_Zmax;                  // диапазон значений Z
        double m_XYTolerance;                 // абс.погрешность задания X и Y
        double m_ZTolerance;                    // абс.погрешность задания Z

        int m_BorderLength;             // используется доп.гран.точек (4|8|16)
        bool m_BorderCreated;             // флаг расчета области определения
        bool m_TriansInited;            // флаг построения нач.тр-ции

        vector<int> m_TrianQue;         // очередь перестраиваемых тр-ков
        vector<int> m_PointQue;         // очередь обрабатываемых точек
        vector<int> m_VertConnect;      // связи точек с вершинами тр-ков

        // предобработка точек и начальная триангуляция
        int PointPreprocessing();
        int InitialTriangulation();
          //функции добавления новой высотной отметки
        int addNewHeight(int point,int manual);
          //функция опеределения правильности задания высоты точки
        int checkHeight(int point);
        //функция интерполяции высоты на отрезке линии по двум известным точкам
        int interpolate(int begin, int end,CLine *line,int beg_index,int end_index);

public:

        CDEMData();
        ~CDEMData();

        // выделение памяти для наборов
        int AllocPoints(int npnt, int lbord = 0);
        int AllocLines(int nlin);

        // создание точек, линий, тр-ков
```

```
CPoint* CreatePoint(double x, double y,
 double z = CPoint::EMPTY_HEIGHT, unsigned char type = 0);
CLine* CreateLine(unsigned char type = 0);
CTrian* CreateTrian(int a, int b, int c, int ab, int bc, int ca);
// добавление точки, линии, тр-ка к соответствующему набору
int Append(CPoint *pnt);
int Append(CLine *lin);
int Append(CTrian *tri);
// изменение существующего треугольника
CTrian* SetTrian(int trinum, int a, int b, int c, int ab, int bc, int ca);
// расчет границ и параметров области определения
int CalcRegionBorder();

// функции и свойства для доступа к элементам наборов
int GetNumPoints();
int GetNumLines();
int GetNumTrians();
CPoint* GetPoint(int pntind);
CLine* GetLine(int linind);
CTrian* GetTrian(int triind);
// очистка наборов
void ClearLines();
void ClearTrians();
void ClearDataSets();
// добавление новой точки к тр-ции Делоне, построение тр-ции Делоне
int AddDelaunayVertex(int newpnt, int begtri);
int DelaunayTriangulation();
//функция проверки и восстановления пропущенных значений высот
//данная функция запускается в момент нажатия пользователем на меню “Расчёт высот 3-D линий”
//включает в себя вызовы функций addNewHeight, checkHeight, interpolate
int hightCheck();
int removeETrian();

};
```

В данной дипломной работе в классе CDEMData были реализованы 4 функции addNewHeight, checkHeight, interpolate, removeTrian.

Функция addNewHeight(int point, int manual) вызывается в методе heightCheck() и служит для добавления новой высотной отметки в модель рельефа. Параметр point задаёт номер точки в наборе точек, с которой должна совпадать новая высотная отметка. Если параметр manual равен нулю, то новая высотная отметка будет добавлена только в случае, если у точки с номером point и любых совпадающих с ней не определена высота. Иначе новая высотная отметка всё равно будет добавлена и её высотой станет высота которая уже определена в точке.

Функция checkHeight(int point) вызывается в методе heightCheck() и служит для проверки правильности задания высоты в точке. Параметр point задаёт номер точки в наборе точек. Если в точке и во всех точках совпадающих с данной не задана высота, то возвращает false, иначе если хотя бы в одной точке высота задана, то возвращаемое значение true. Если значения высот в точках различны, то возвращаемое значение false.

Функция checkHeight() выявляет линии с незаданной высотой, при необходимости добавляет новые точки в модель рельефа и производит расчёт высот незаданных вершин линий. Данная функция вызывается в ответ на нажатие пользователем пункта меню

“Расчёт высот 3-D линий”. Возвращает количество точек не в которых необходимо определить высоту для проведения интерполяции. Возвращает ноль если во всех необходимых точках высота задана и интерполяция произведена.

Функция interpolate(int begin, int end, CLine *line, int beg_index, int end_index) производит восстановление значений высот линии line. Параметры begin и end задают глобальные номера точек в наборе точек. Параметры beg_index и end_index задают номера точек в линии.

Функция removeETrian выявляет все недопустимые горизонтальные треугольники и перестраивает их. Эта функция вызывается в ответ на нажатие пользователем пункта меню “Проверка сильных ограничений”. Возвращает количество новых добавленных точек, либо ноль если точек добавлено не было.

Класс CLine - базовый класс для линий поверхности.
Объекты содержат номера начальной и конечной вершин линии
в связанном с линией наборе точек (линия хранится в виде списка).

```
class CLine
{
};
```

Класс CPoint - базовый класс для точек поверхности.

```
class CPoint
{
};
```

Класс, используемый для организации наборов точек, линий, тр-ков.

```
template <class T>
class CSet
{
};
```

Класс CShape базовый класс для загрузки шейп-фала с жесткого диска в систему. Класс содержит в себе описатели основных свойств шейп-файла. При выборе пользователем пункта меню “Добавить шейп” создаётся новый объект класса CPointShape или CLineShape

```
class CShape
{
protected:
        AnsiString m_FileName;                  // полное имя файла
         int m_Type;                  // тип шейпа (1 или 3)
        unsigned short m_DBHeadLen;     // длина заголовка DBF
        unsigned short m_DBRecLen;      // длина записи DBF
        int m_DBFieldNum;               // число полей в записи DBF
        int m_HeightOffset;             // смещение поля HEIGHT от начала записи
        int m_HeightLength;             // длина поля HEIGHT
        char m_HeightFormat[10];        // формат для поля HEIGHT
        unsigned char *m_DBFieldHead;   // буфер заголовков всех полей

public:
       double Xmin, Ymin, Xmax, Ymax;  // мин. и макс. координаты
        int LineNumber;                // число линий шейпа (или 0)
        int PointNumber;               // число точек/вершин линий
        TColor MainColor;              // цвет исходных точек/изолиний
        TColor AddColor;               // цвет 3D-линий/добавленных точек
        bool Changed;                  // флаг изменения точек/линий шейпа

        CShape();
```

```
        virtual ~CShape();

        // формирование описателя CPointShape или CLineShape
         static CShape *CreateShape(AnsiString FName);

         // свойства для чтения значений закрытых переменных
           __property AnsiString FileName = { read = m_FileName };
           __property int Type = { read = m_Type };

           virtual int Load(CDEMData *InData);
           virtual int Save(CDEMData *OutData);
};
```

Классы наследующиеся от класса CShape и перегружающие функции Load и Save для загрузки определённых типов шейпов.

```
class CPointShape : public CShape;
class CLineShape : public CShape;
```

Класс CTrian представляет треугольники.

```
class CTrian
{
};
```

Класс CProject представляет собой базовый класс для управления всей системой классов и предоставления пользовательского интерфейса. Включает в себя наборы шейпов загруженных в проект, методы работы с уже загруженными шейпами. Предоставляет набор методов по работе с проектом, как то создать новый, сохранить текущий, сохранить текущий как. Содержит методы для загрузки шейпов добавленных в проект в основной класс обработки данных CDEMData

```
class CProject
{
private:
        CSet<CShape> m_Shapes;      // набор указателей для всех шейпов проекта
        CDEMData *m_ProcData;       // объект, хранящий информацию из шейпов

public:
         AnsiString ProjectName;     // имя файла для сохранения проекта
        bool Changed;               // флаг изменений в проекте

        CProject();

         // установка связи проекта с хранилищем данных
         void SetDEMData(CDEMData *Data);


        CShape *GetShape(int shpnum);
        int GetShapeNumber();
        __property CShape* Shape[int shpnum] = { read = GetShape };
        __property int ShapeNumber = { read = GetShapeNumber };

         int ClearProject();                    // очистка текущего проекта
         int OpenProject();                     // открытие существующего проекта
        int AddToProject(CShape *shp);  // добавление нового шейпа к проекту
         int LoadData();            // загрузка данных из шейпов проекта
         int SaveData();           // сохранение измененных шейпов проекта
```

```
        int QuerySaveData();        // запрос + сохранение измененных шейпов
        int SaveProject();          // сохранение dep-файла проекта
        int QuerySaveProject(int Query = 1); // запрос+сохранение файла проекта
};
```

Таким образом в данной дипломной работе были реализованы несколько методов класса CDEMData и предложен и реализован класс CProject

Заключение

При выполнении дипломной работы были получены следующие результаты:

1. Разработан и реализован алгоритм восстановления высот в узлах трехмерных структурных линий рельефа.
2. Разработан и реализован алгоритм построения триангуляции с сильными ограничениями.
3. Реализована система классов для формирования проекта.

Список использованных источников

1. Скворцов А.В. Триангуляция Делоне и ее применение. – Томск: Изд-во Том. ун-та, 2002. – 128 с.
2. Препарата Ф., Шеймос М. Вычислительная геометрия: Введение. Пер. с англ. – М.: Мир, 1989. – 478 с.
3. Костюк Ю.Л., Фукс А.Л. Предварительная обработка исходных данных для построения цифровой модели рельефа местности // Вестник ТГУ, 2003. №280. С. 281-285.
4. Костюк Ю.Л., Фукс А.Л. Построение цифровой модели рельефа местности на основе структурных линий и высотных отметок // Вестник ТГУ, 2003. №280. С. 286-289.
5. Архангельский А.Я. Программирование с C++ Builder 6. – М: ЗАО «Издательство БИНОМ», 2004. – 1152 с.
6. ESRI Shapefile: A Technical Description. Environmental System Research Institute Inc., USA, 1997. – 23 p.
7. ARC/INFO Data Model, Concepts, and Key Terms. The geographic information system software. Environmental System Research Institute Inc., USA, 1992. – 320 p.