Лабораторная работа №2
по курсу "Алгоритмы вычислительной геометрии"

Построение и анализ триангуляции Делоне

Самохина Алина, 904а

18 декабря 2019 г.

1 Постановка задачи

Дано: Файл исходных данных: облако из n точек на плоскости. Точки изображают фигуру.
Требуется: Оценить периметр и площадь фигуры на основе аппроксимации её границы многоугольником.
Программа для решения задачи должна выполнять:

1. Ввод заданного массива точек
2. Построение триангуляции Делоне
3. Удаление "длинных" рёбер, чья длина перевышает заданный порог
4. Построение оболочки-огибающей оставшегося подграфа в виде многоугольника
5. Вычисление и вывод периметра и площади полученного многоугольника
6. Визуализацию исходного множества точек и полученной оболочки

1.1 Описание данных

Исходные данные задаются в текстовом файле. Первая запись – число точек, далее координаты точек, разделенные пробелом. Координаты точек – целые числа в диапазоне размеров экрана компьютера.

1.2 Выходные данные

На выходе программа должна показать: имя файла исходных данных, периметр и площадь построенной фигуры, изображение облака точек и многоугольной границы.

2 Алгоритм решения

2.1 Описание алгоритма

В рамках данной задачи рассматривается несколько подзадач:

1. Построение триангуляции Делоне
2. Удаление "длинных" рёбер из триангуляиции
3. Построение огибающей получившегося подграфа
4. Вычисление периметра и площади

2.1.1 Построение триангуляции Делоне

Построение триангуляции Делоне было выполнено с использованием библиотеки scipy.spatial для Python. Данная реализация использует библиотеку QHull и, соответственно, следующий алгоритм построения триангуляции Делоне:

1. Точки, поданные на вход, отображаются на параболоид с использованием суммы квадратов координат

2. В трёхмерном пространстве строится выпуклая оболочка полученных точек с помощью алгоритма quickhull (алгоритм быстрой оболочки)

3. Нижние грани выпуклой оболочки проецируются на исходную плоскость, образуя триангуляцию Делоне

Сложность данного алгортма в среднем - $O(n \log n)$, по сложности алгоритма построения выпуклой оболочки QuickHull.

2.1.2 Удаление "длинных"рёбер

В данной лабораторной работе удаление "длинных" рёбер реализовано следующим образом. Программа проходит по массиву треугольников построенной триангуляции. В новую триангуляцию без "длинных" рёбер попадают те треугольники, в которых все рёбра не превышают по длине $\mu + 1.5\sigma$, где μ - средняя длина ребра по всем треугольникам, а σ - стандартное отклонение. Данное значение было получено эмпирически. Сложность операции O(n), так как Эйлером было доказано, что количество треугольников в триангуляции Делоне - O(n).

Data: массив всех треугольников триангуляции Делоне - tri
Result: массив треугольников триангуляции, не содержащих "длинных"рёбер - tri_new

```
for треугольник ∈ tri do
    Подсчёт длины каждого из трёх рёбер
    if каждое ребро < μ + 1.5σ then
        добавить треугольник → tri_new
    end
end
```

Algorithm 1: Удаление длинных рёбер

2.1.3 Построение огибающей подграфа в виде многоугольника

Данная операция также выполняется за O(n). Алгоритм проходит по массиву треугольников новой триангуляции, составленной на предыдущем шаге. Составляет список рёбер и количество их вхождений в триангуляцию. В следующем проходе по массиву рёбра, встречающиеся в триангуляции единожды, добавляются в огибающую подграфа.

Data: массив треугольников триангуляции, не содержащих "длинных" рёбер - tri_new
Result: Массив рёбер огибающей подграфа - hull
инициализировать словарь рёбер edges **for** *треугольник* $\in$ *tri* **do**
  for *ребро* $\in$ *треугольник* **do**
    if *ребро* $\notin$ *edges* **then**
      добавить ребро $\rightarrow$ edges
      edges[ребро].количество := 1
    else
      edges[ребро].количество := edges[ребро].количество + 1
    end
  end
end
for *ребро в edges* **do**
  if *edges[ребро].количество = 1* **then**
    добавить ребро в hull
  end
end

Algorithm 2: Построение огибающей

2.1.4 Вычисление площади и периметра

Периметр фигуры считается как сумма рёбер огибающей подграфа.
Сложность - O(n)
Площадь фигуры считается как сумма треугольников триангуляции без "длинных"рёбер.
Сложность - O(n)

3 Инструкция по работе с программой

1. Запустить скрипт main.py/jupyter-notebook main.ipynb
2. Выбрать картинку, с которой будет работать программа:
   - Выбрать рыб/птиц с помощью нажатия "f"/"b"
   - Из предложенного диапазона выбрать номер картинки и ввести его
3. Появляется окно с визуализацией результатов
4. После закрытия окна визуализации можно заново выбрать картинку, нажав "1" или завершить работу с программой.

4 Выводы

В данной работе были реализованы алгоритмы, позволяющие построить триангуляцию Делоне и с её помощью найти контур фигуры, а также подсчитать её площадь и периметр за O(nlogn), где n - количество точек в файле входных данных.

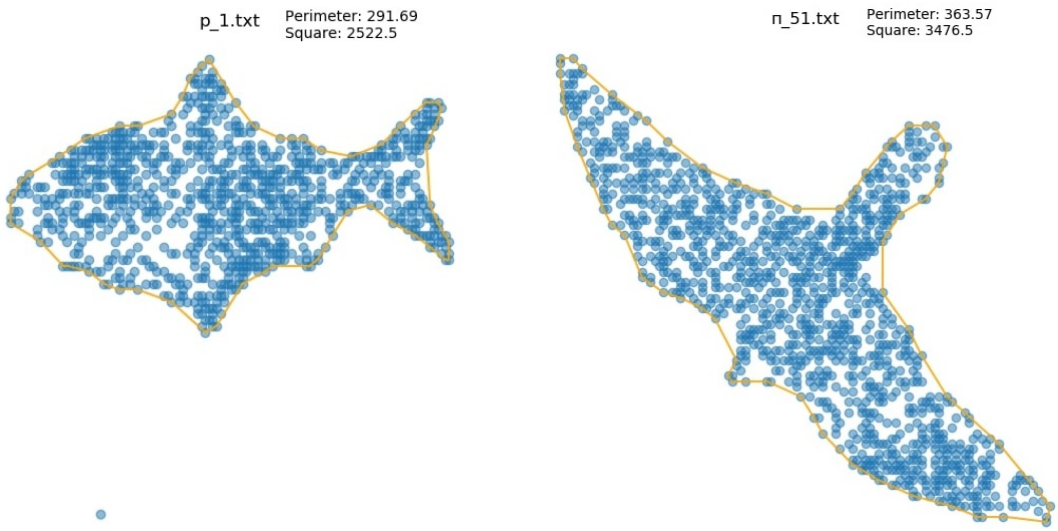

Рис. 1: Примеры работы программы