Работа с выпуклыми многоугольниками

Степанов Всеволод
Практика, весна 2015
Куратор: Сергей Копелиович

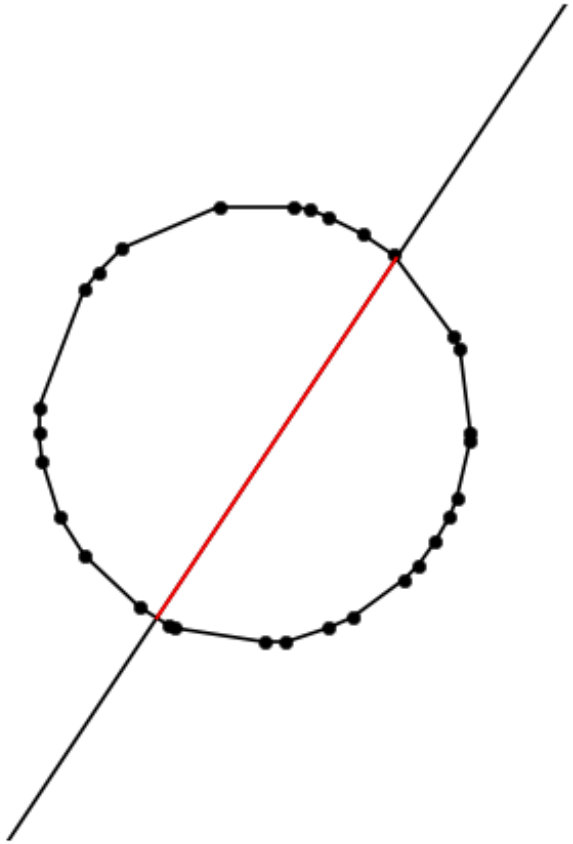
Цели и задачи:

Есть ряд алгоритмов для выпуклых многоугольников, которые можно реализовать за O(log n), все они были реализованы:

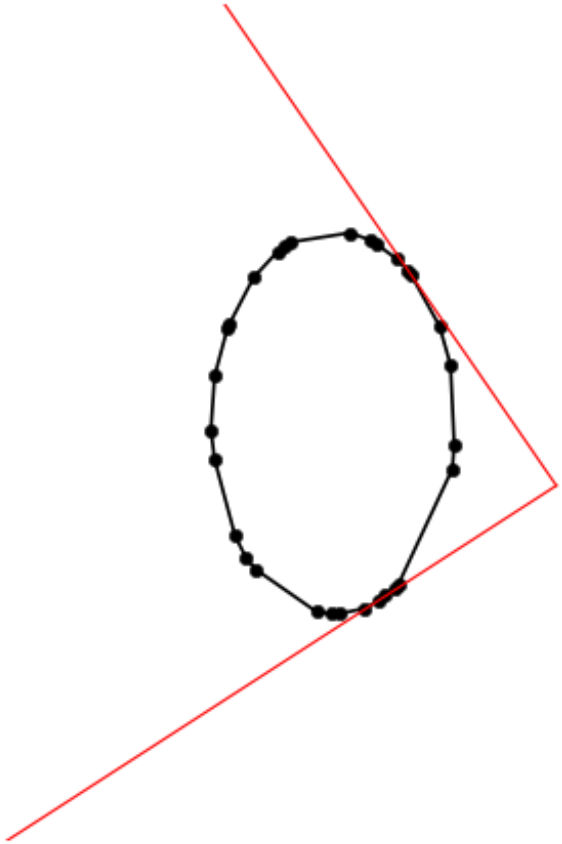
- Проверка на принадлежность точки многоугольнику
- Поиск касательных из точки
- Расстояние от точки до многоугольника
- Пересечение многоугольника с прямой
- Поиск точки максимально удаленной по данному направлению

Отсутствие библиотек для C++, в которых эти алгоритмы были бы реализованы

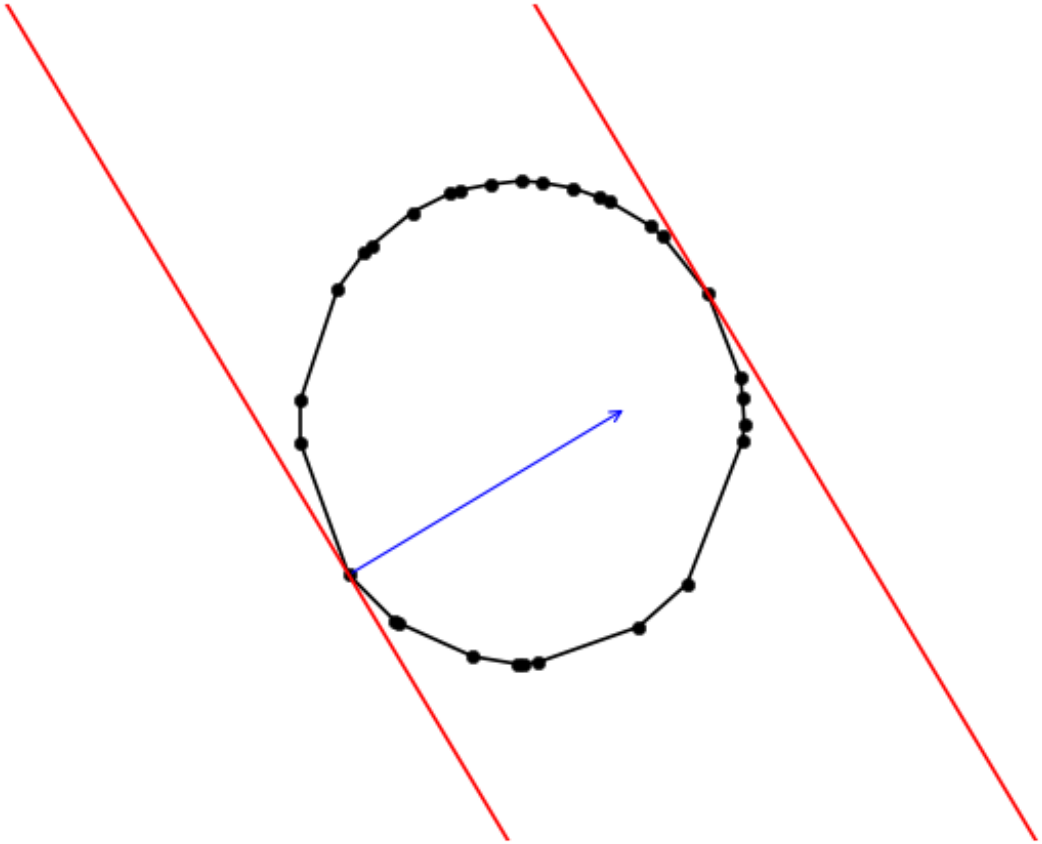
Пересечение прямой и многоугольника



Касательные



Поиск экстремальных точек по направлению



Цели и задачи:

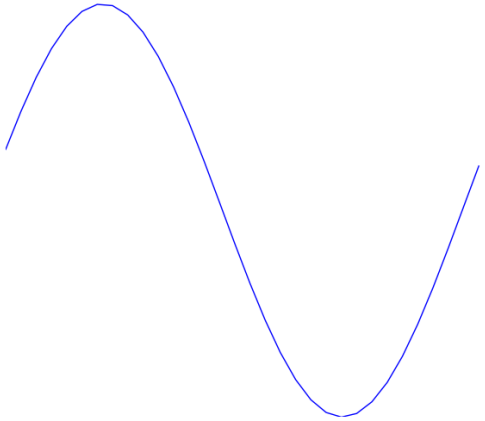
Визуализация этих алгоритмов.

- Удобно для отладки
- Можно наглядно объяснить работу алгоритма
- Желательно, чтобы визуализатор можно было использовать и для отображения каких-то других данных

Реализация

- Язык: C++
- 5 алгоритмов
- К каждому алгоритму есть наивная реализация
- Ручные тесты, а так же генератор больших прямоугольников

Алгоритмы



- Все алгоритмы основаны на бинпоиске по какой-либо функции (например, по углу)
- Как правило, функция не немонотонна
- Крайние случаи, которые надо обрабатывать
- Все делается в целочисленной арифметике, чтобы избежать проблем с точностью

Реализация: тестирование

- Для каждого алгоритма есть свой набор тестов, проверяющий его работу в том числе и на различных крайних случаях
- Реализованы наивные алгоритмы и генератор больших случайных выпуклых многоугольников, на них результат работы быстрого алгоритма сравнивался с результатом медленного

Реализация: визуализация

C++:

- Можно делать визуализацию прямо из кода
- Нету стандартной графической библиотеки
- Сторонние библиотеки: зависимость алгоритмической части от их наличия

Реализация: визуализация

Python:

- Алгоритмическая и графические части независимы
- Можно передавать команды откуда угодно
- Есть стандартная графическая библиотека
- И много сторонних удобных модулей

Визуализация: метаязык

- Читаемый формат
- Можно пользоваться визуализатором

segment -1607 624 -1712 -233 color=black

point -1607 624

point -15 -20 name=point

print Проверка на принадлежность точки многоугольнику

Стандартная библиотека

TKinter

- Подходит для создания простенького приложения с GUI
- Есть Canvas, на котором можно рисовать графические примитивы
- Детали (масштабирование, чуть более сложные объекты) ложатся на программиста
- Без сторонних библиотек нельзя сохранить изображение

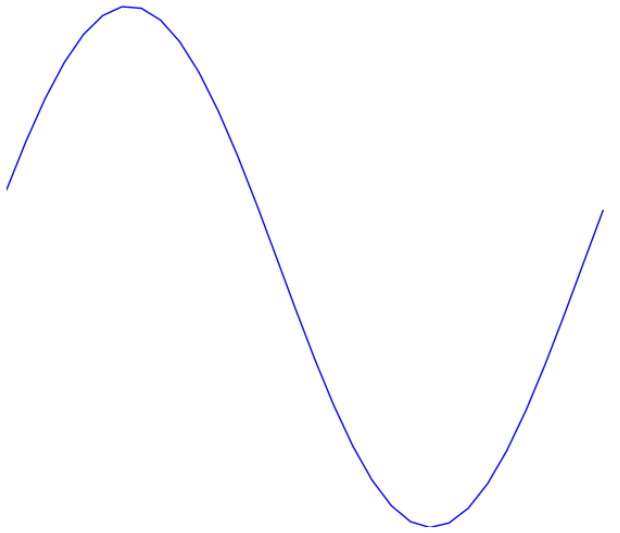
Matplotlib

- Библиотека для отрисовки разнообразных графиков и не только
- Возможность работы с GUI
- Не надо заботиться о деталях
- Возможность сохранять нарисованное на экране в изображение или в видео

Matplotlib

Все действительно очень легко и удобно

```
import matplotlib.pyplot as plt
import numpy as np
x = np.arange(0, 2*np.pi, 0.2)
y = np.sin(x + 0.3)
plt.plot(x, y)
plt.savefig('sin.png')
```



Что еще можно сделать

Есть алгоритм поиска расстояния между двумя выпуклыми многоугольниками, работающий за $O(\log n + \log m)$, который можно написать