

# Курсовая работа на тему: «Булевы операции над полигональными моделями»

Выполнили: студенты группы РК9-72

Кобзарь В. А.

Романов А. С.

Руководитель: Хрыков С. С.

# Цели

Разработать программу, предоставляющую пользователю возможность выполнения булевых операций (объединение, пересечение, вычитание) над полигональными моделями.

# Задачи

1. Обзор и анализ существующих решений
2. Разработка алгоритма реализации булевых операций
3. Написание программы, реализующей выбранный алгоритм
4. Тестирование программы

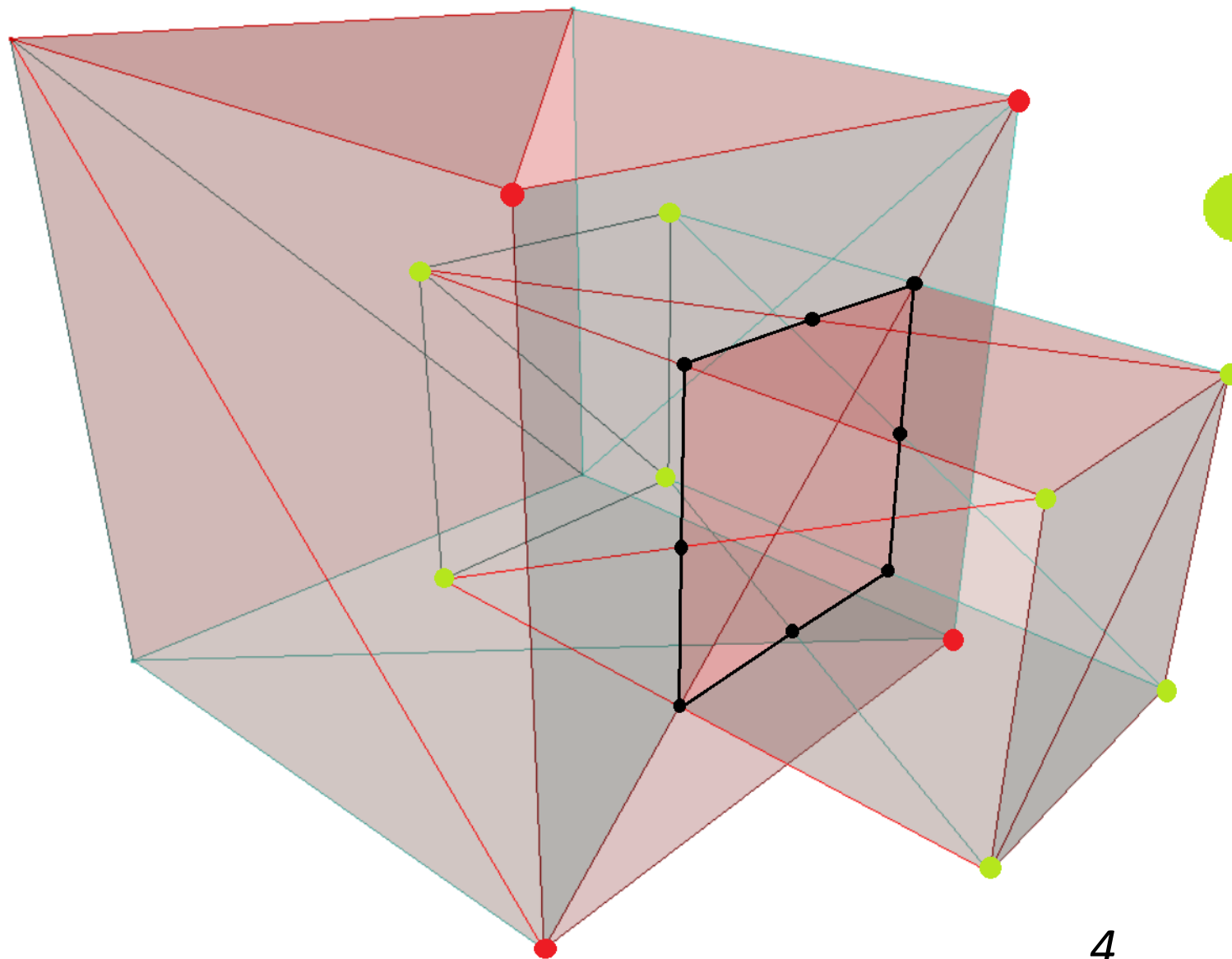
# Обзор и анализ существующих решений

Веб – приложений с возможностью выполнения булевых операций не найдено

## Разработка алгоритма реализации

1. Поиск линии пересечения двух моделей
2. Триангуляция моделей в областях пересечения
3. Разбиение моделей на внутренние и внешние части
4. Получение результата операции путём сложения частей моделей (в зависимости от операции)

# Поиск линии пересечения двух моделей



● - точки пересечения

● ● - вершины треугольников

Результат пересечения двух  
треугольников записывается в  
Polygon

```
struct Polygon {  
    vector<Point> points;  
    int triangle_index;  
  
    void AddPointsToPolygon(vector<Point>& points);  
};
```

# Подзадачи поиска пересечения

## 1. Предварительная проверка пересечения треугольников

$\pi_2: N_2 \cdot X + d_2 = 0$  - уравнение плоскости

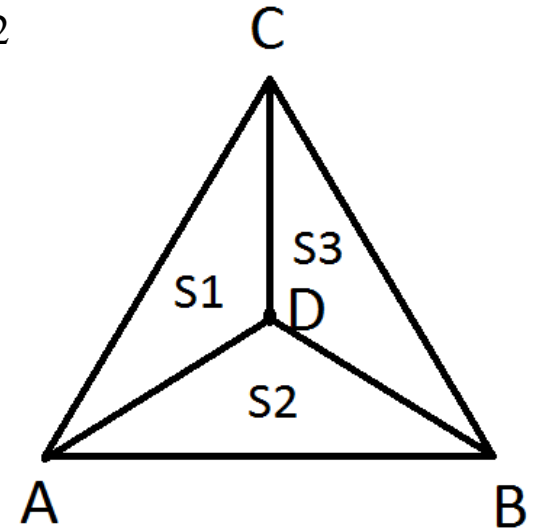
$N_2 = (V_1^2 - V_0^2) \times (V_2^2 - V_0^2)$  - нормаль плоскости  $\pi_2$ .

$$d_2 = -N_2 \cdot V_0^2$$

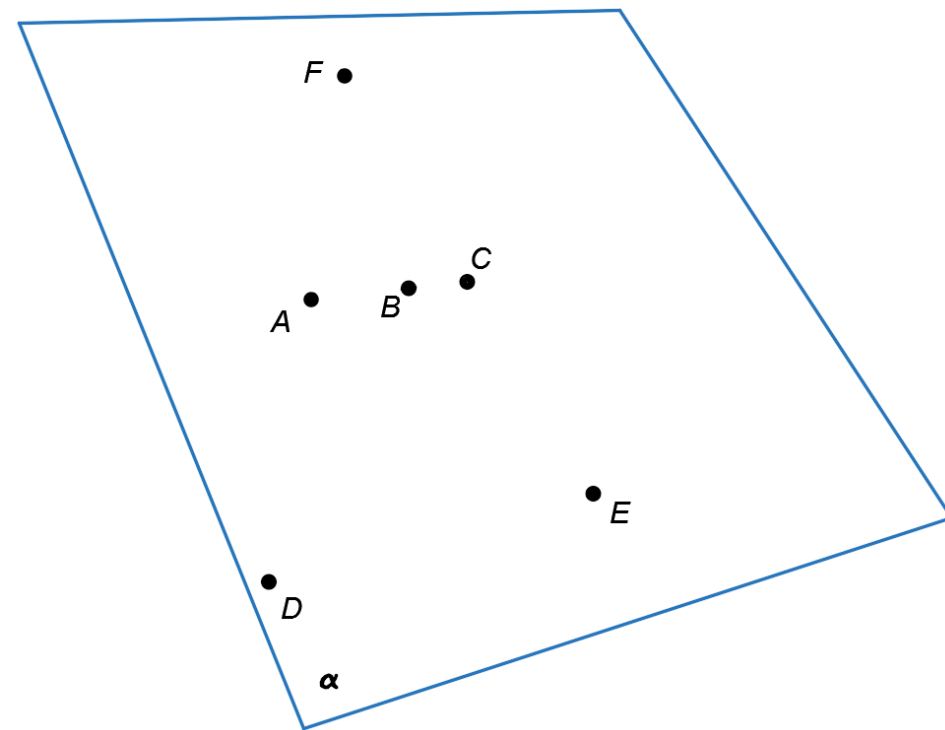
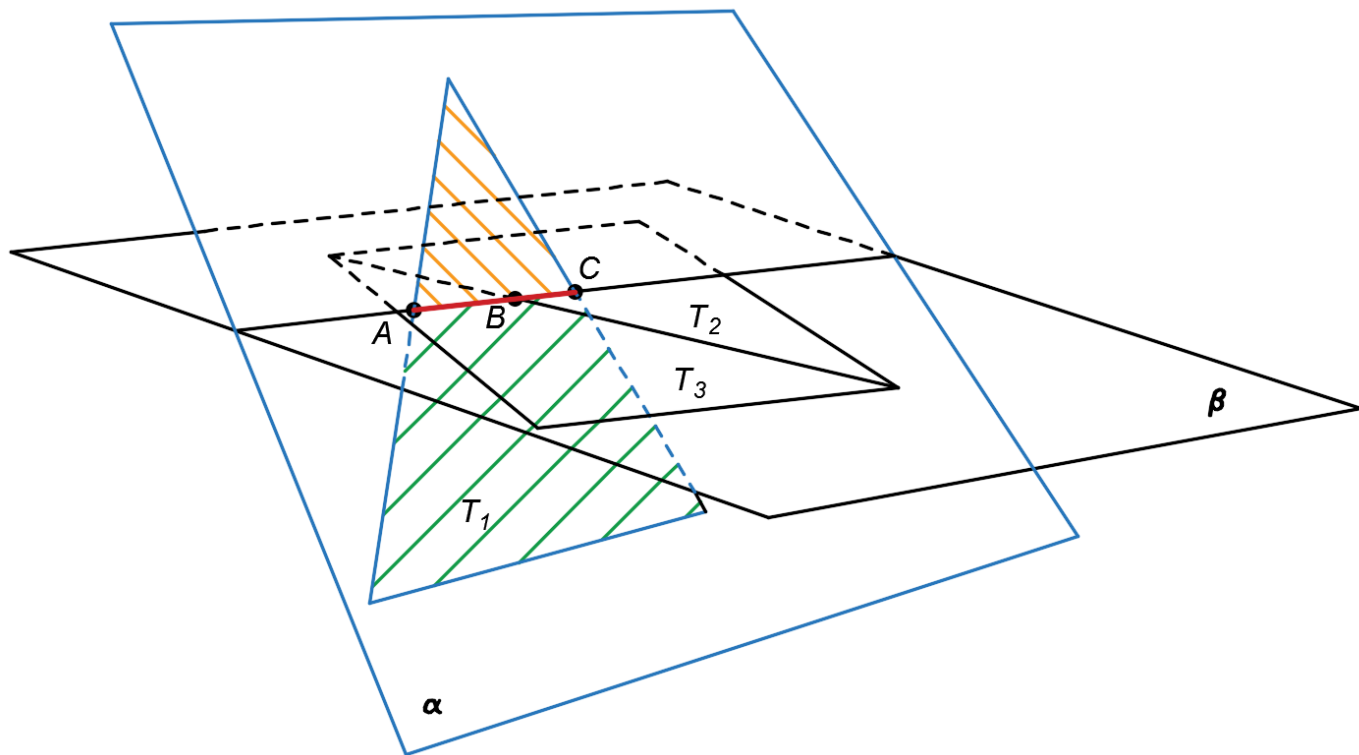
$d_{V_i^1} = N_2 \cdot V_i^1 + d_2, i = 0, 1, 2.$  - расстояния (с учетом знака) от вершин  
треугольника  $T_1$  до плоскости  $\pi_2$

## 2. Проверка принадлежности точки пересечения треугольникам

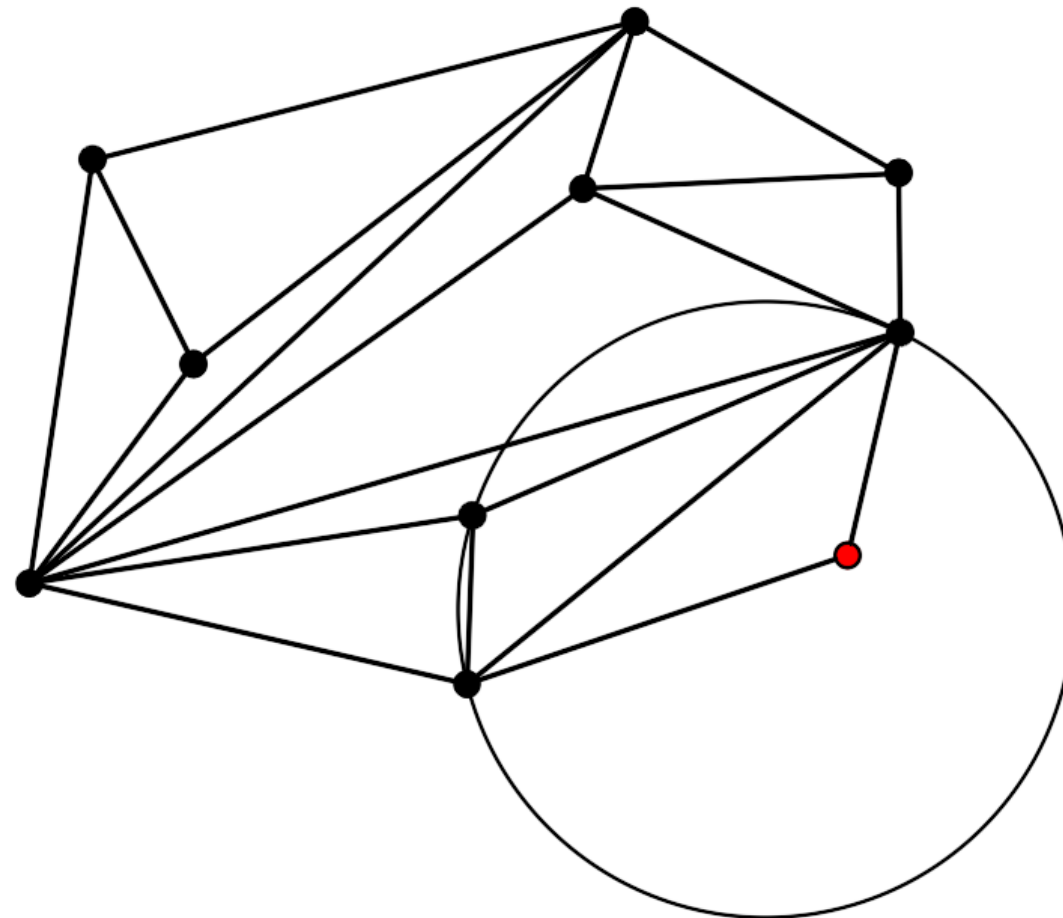
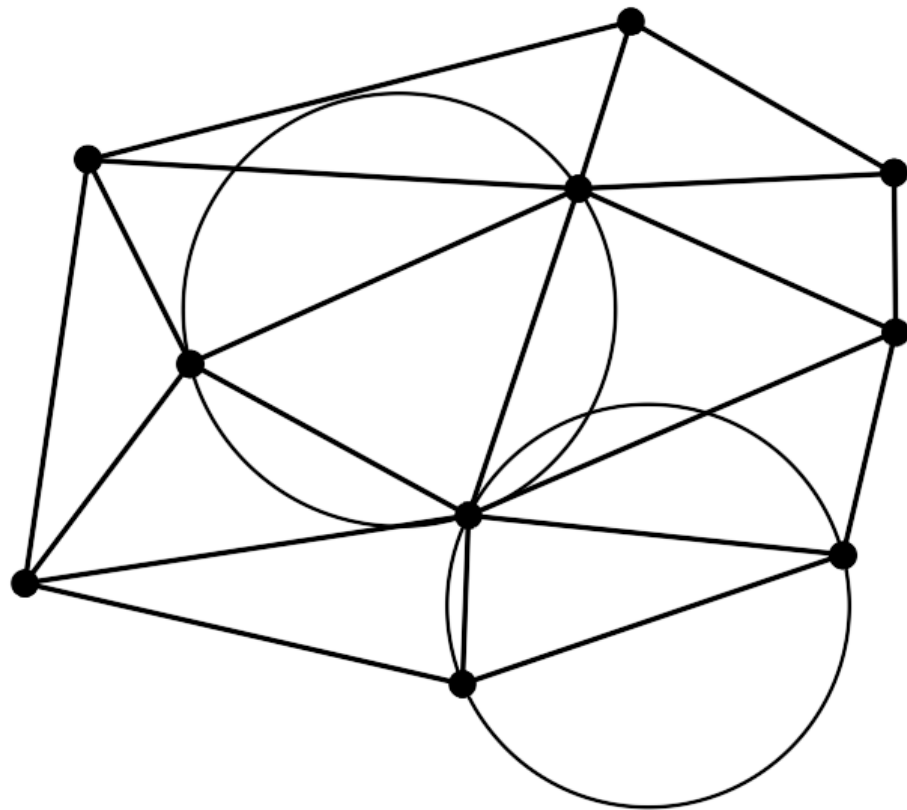
$$S_{ABC} = S_1 + S_2 + S_3$$



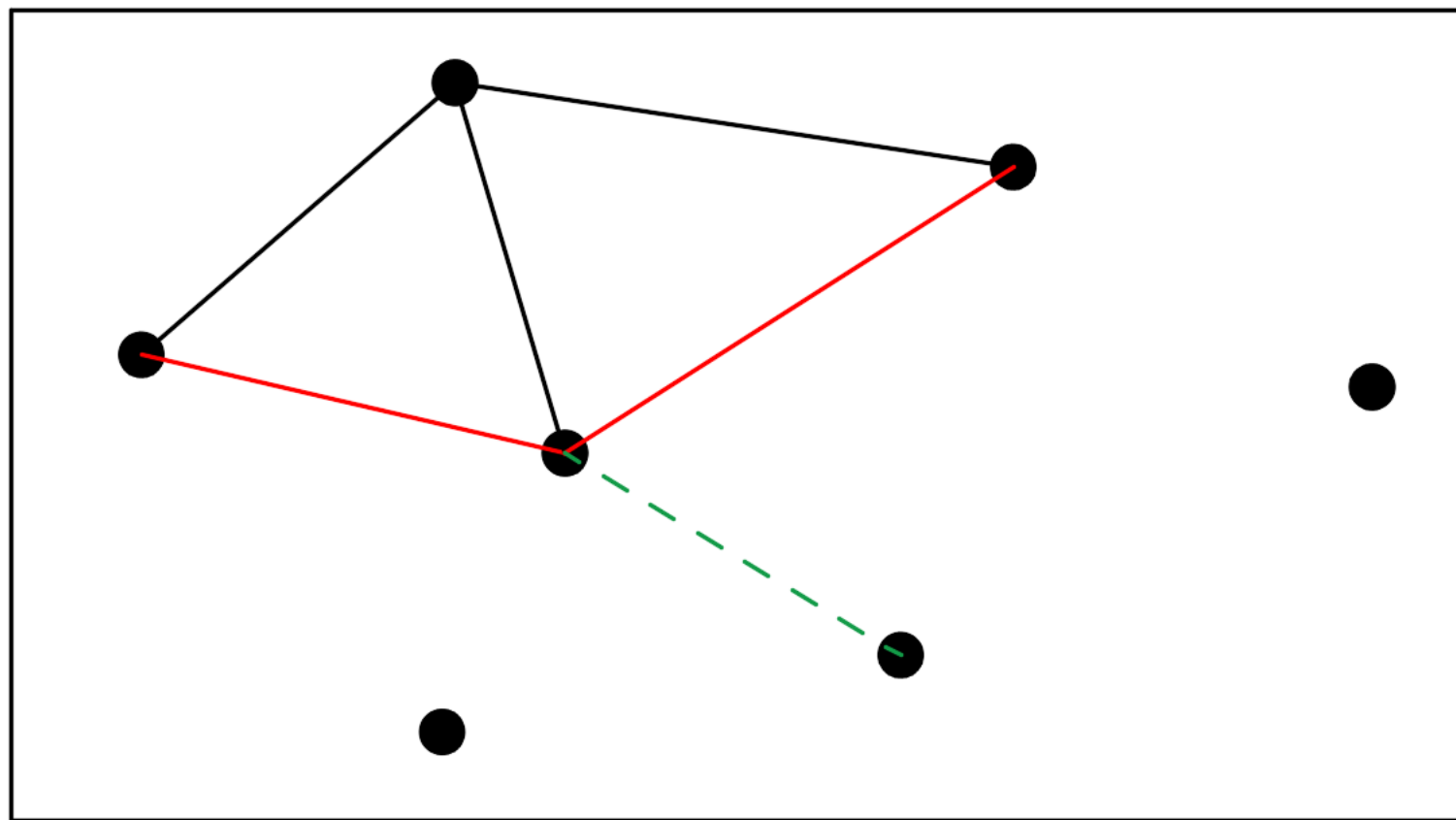
# Триангуляция области пересечения






# Триангуляция Делоне



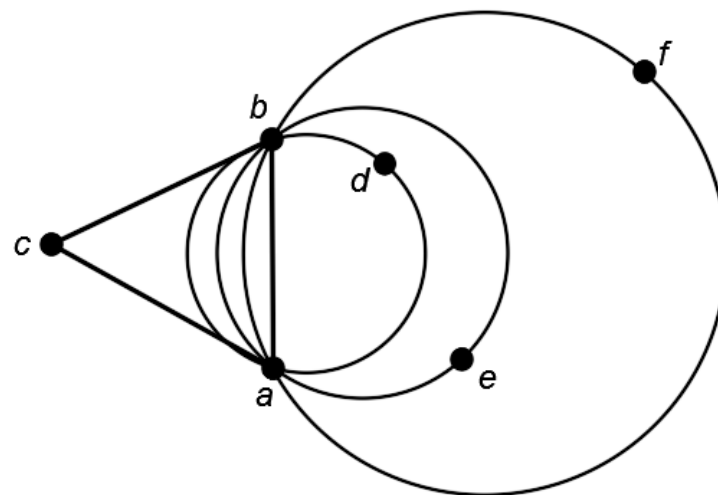
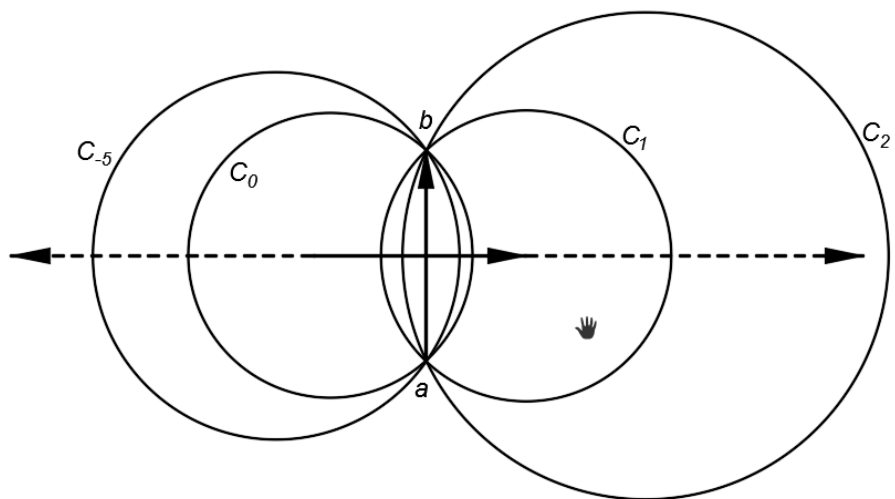
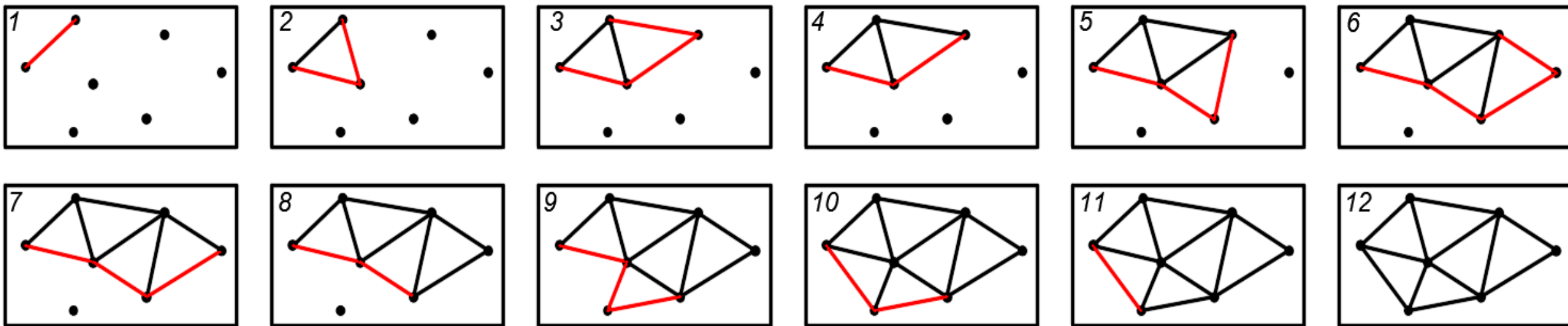
# Инкрементальный алгоритм



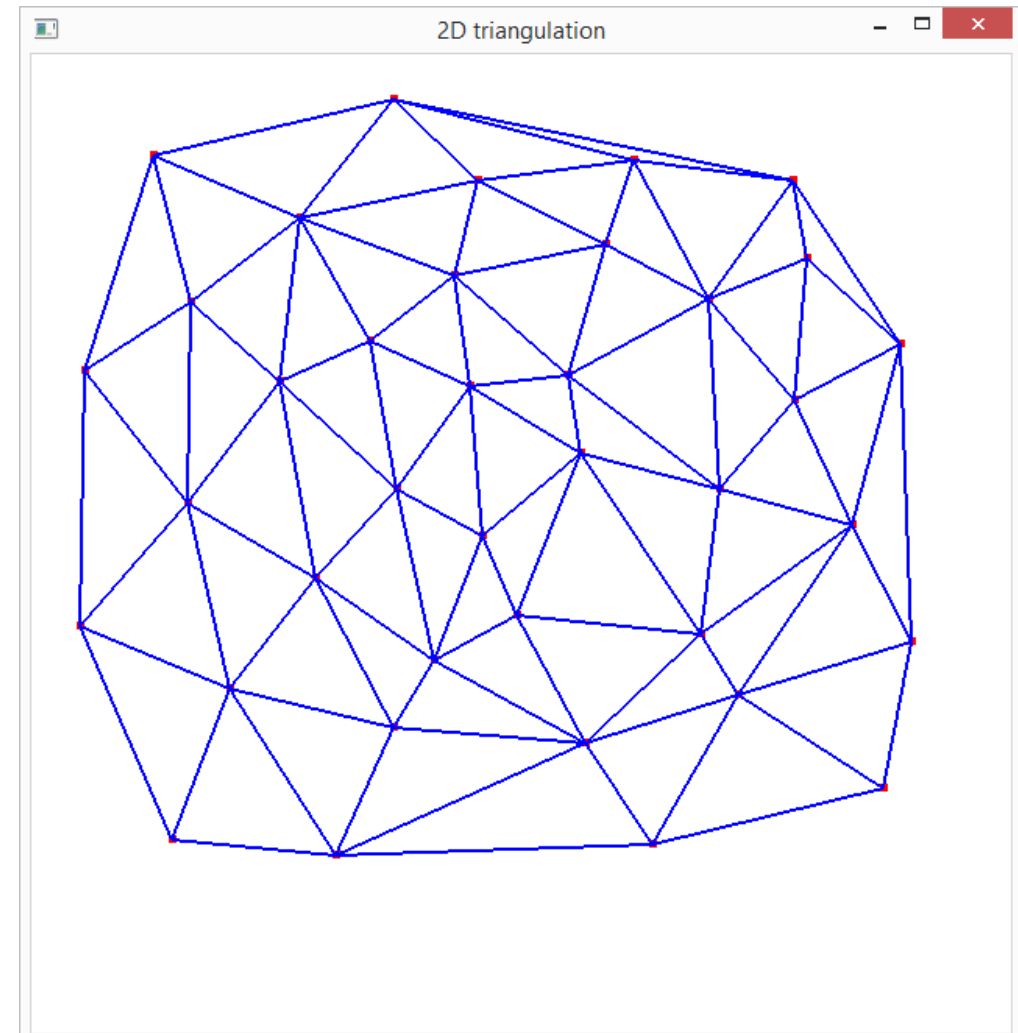
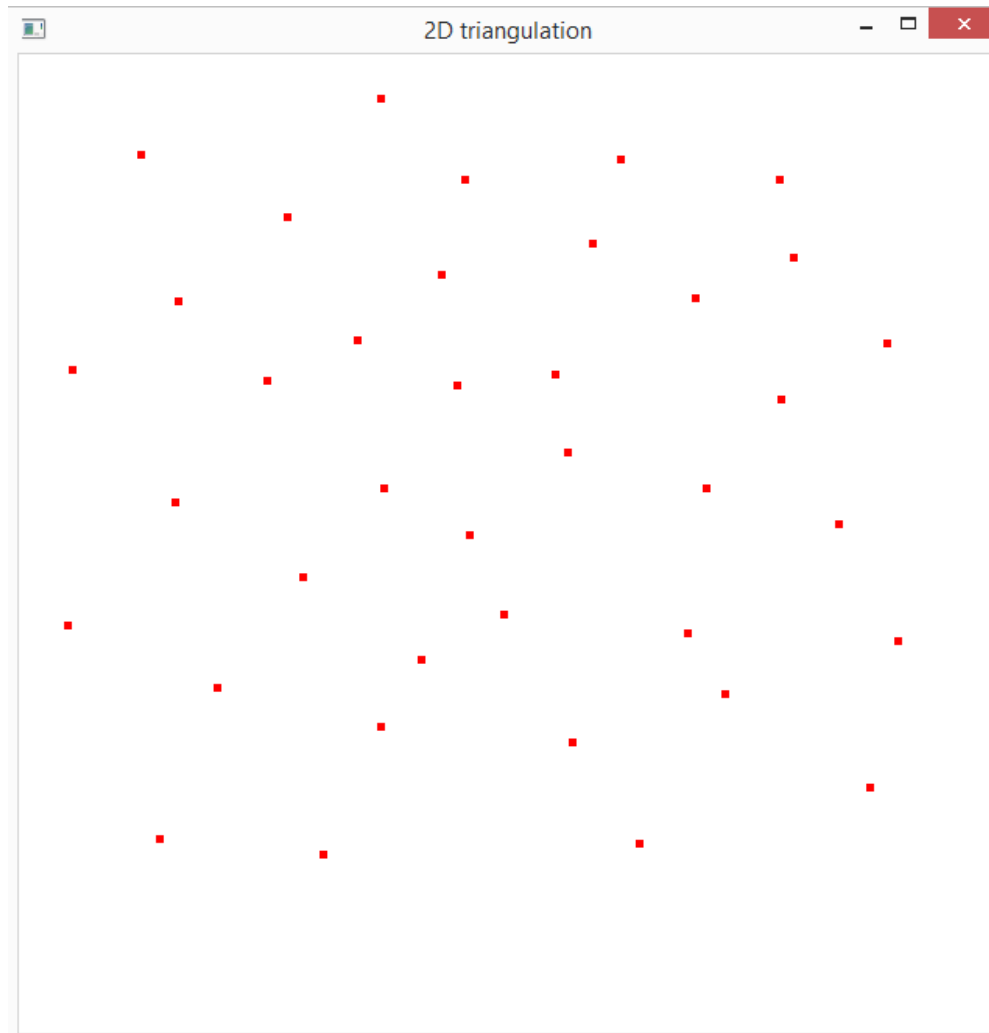
-  -спящее ребро
-  -живое ребро
-  -мертвое ребро



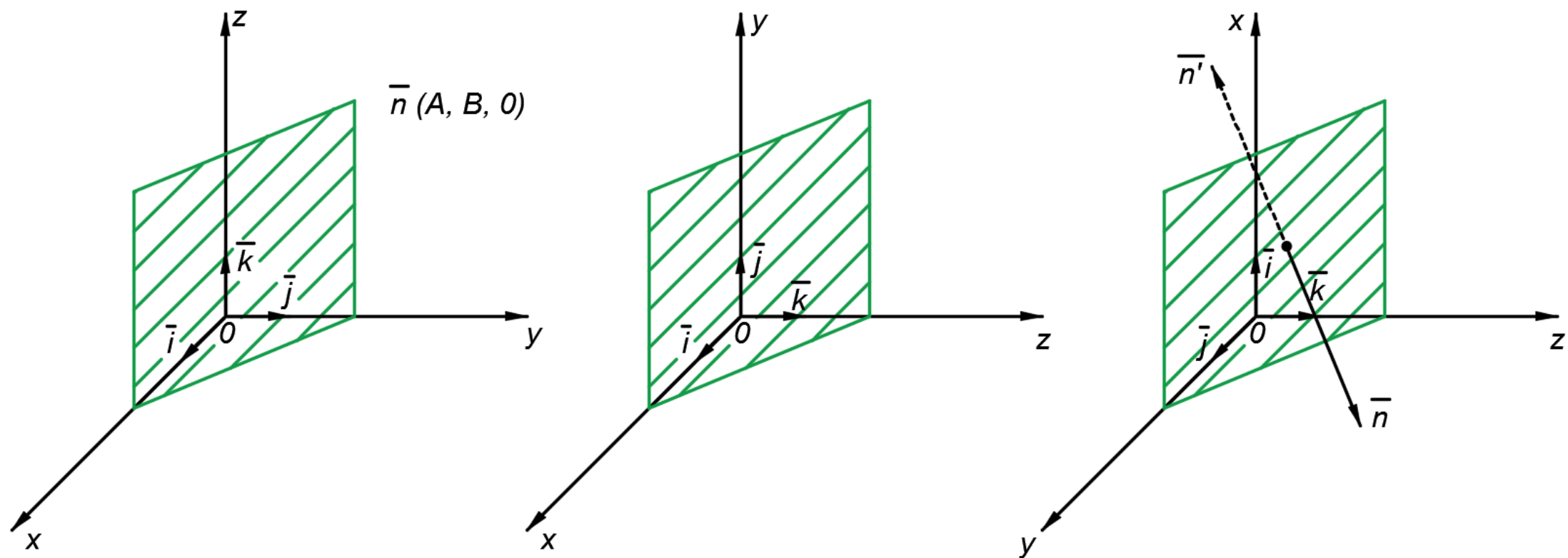
# Инкрементальный алгоритм



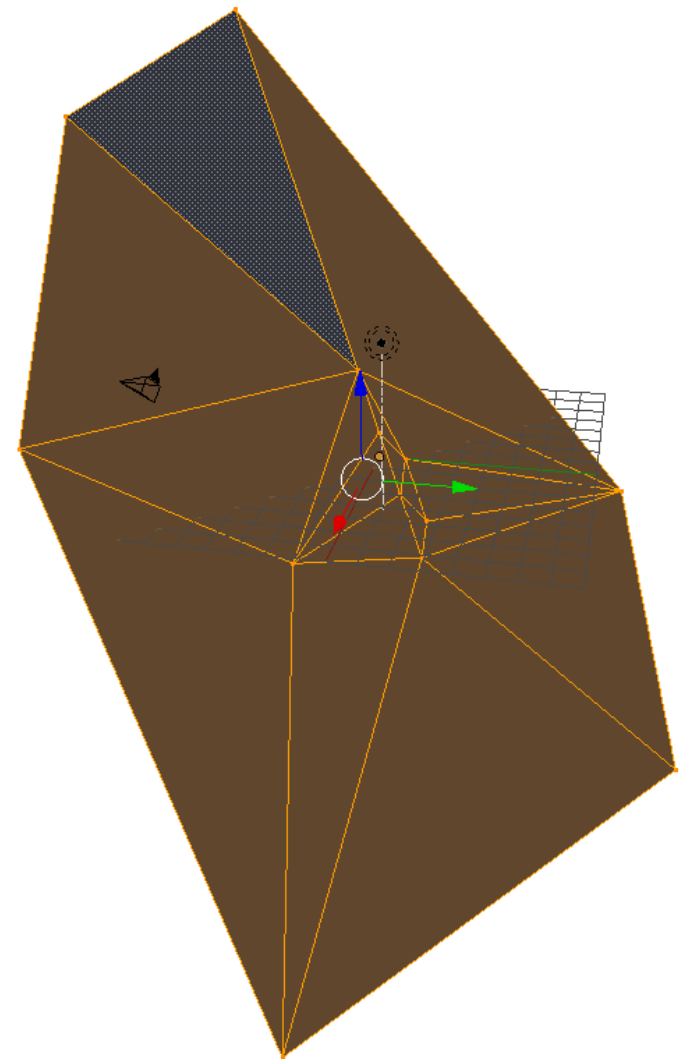
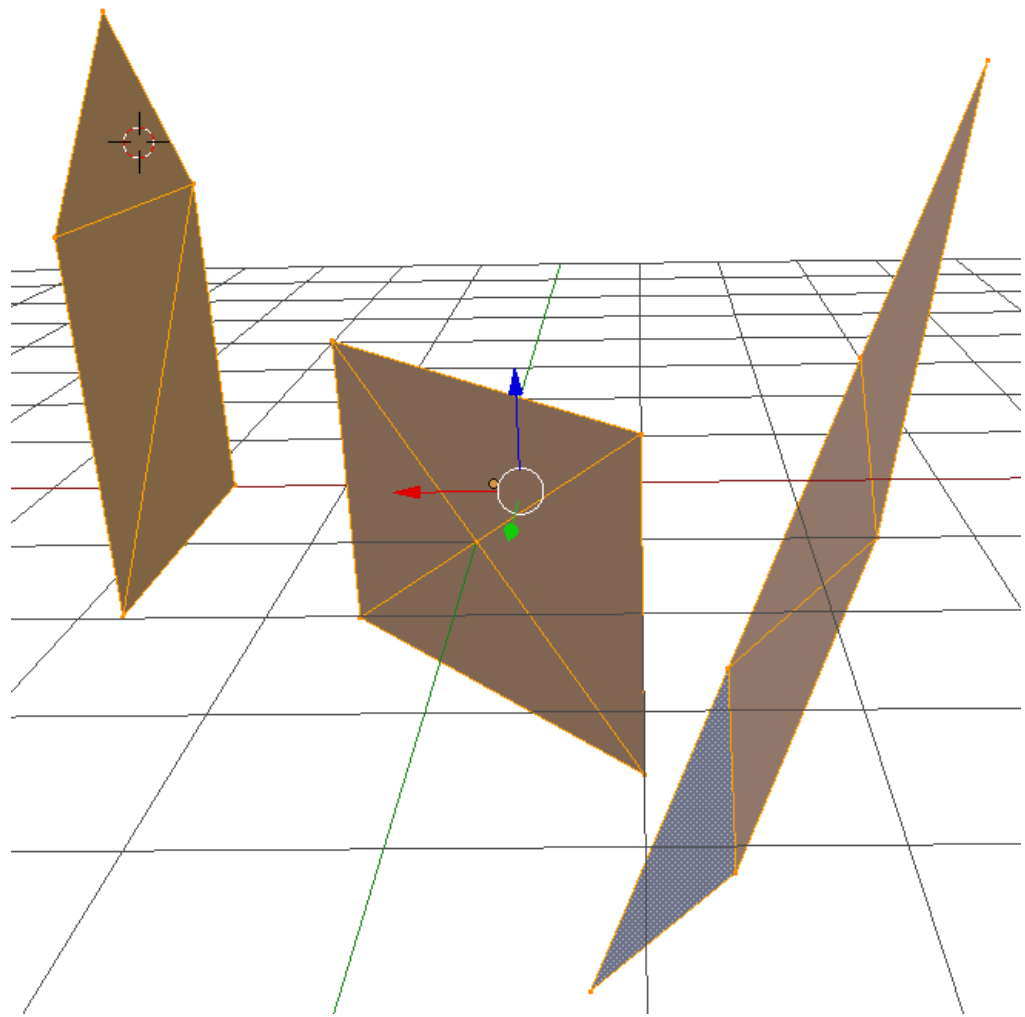
# Результат 2D триангуляции



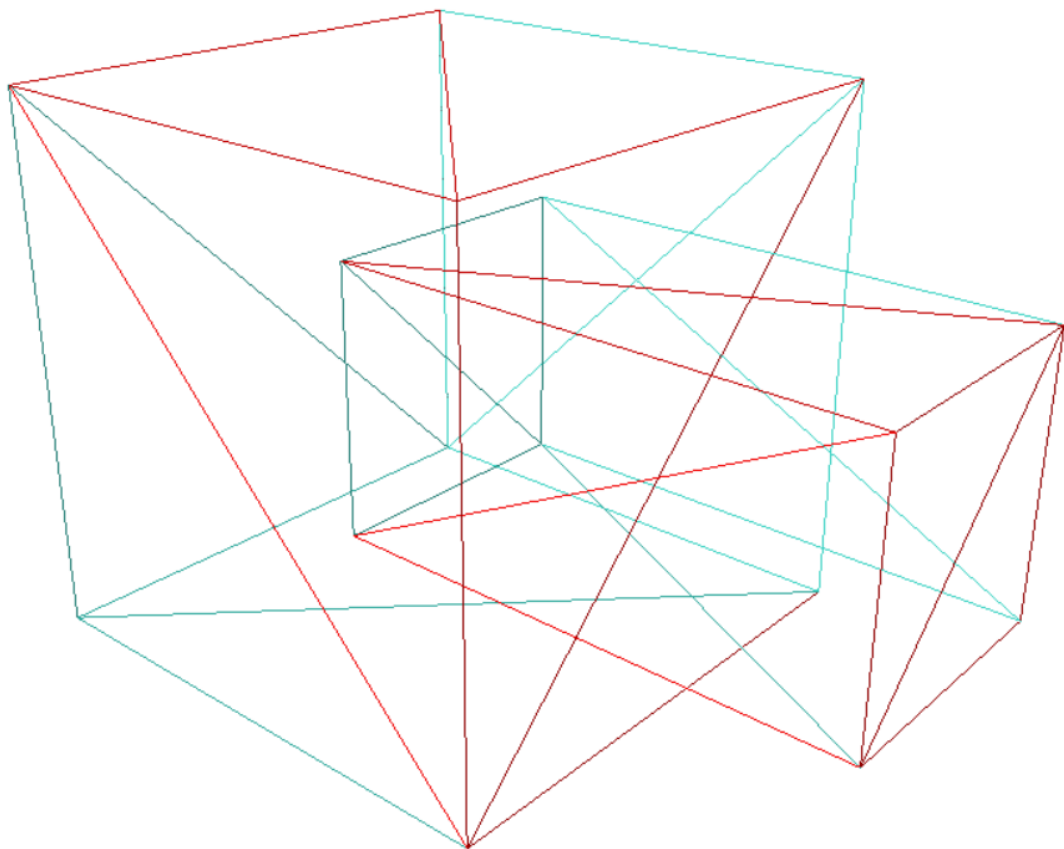
# Разработка пространственного алгоритма триангуляции



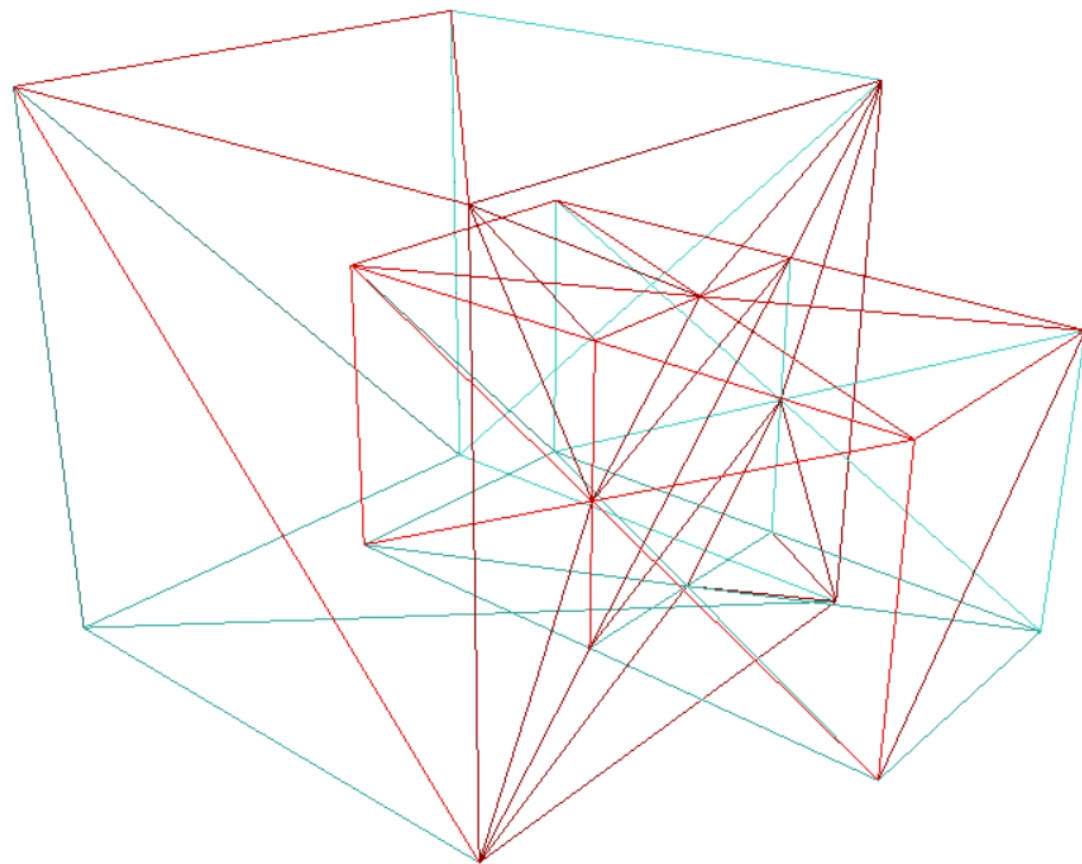
# 3D триангуляция



# Промежуточные результаты

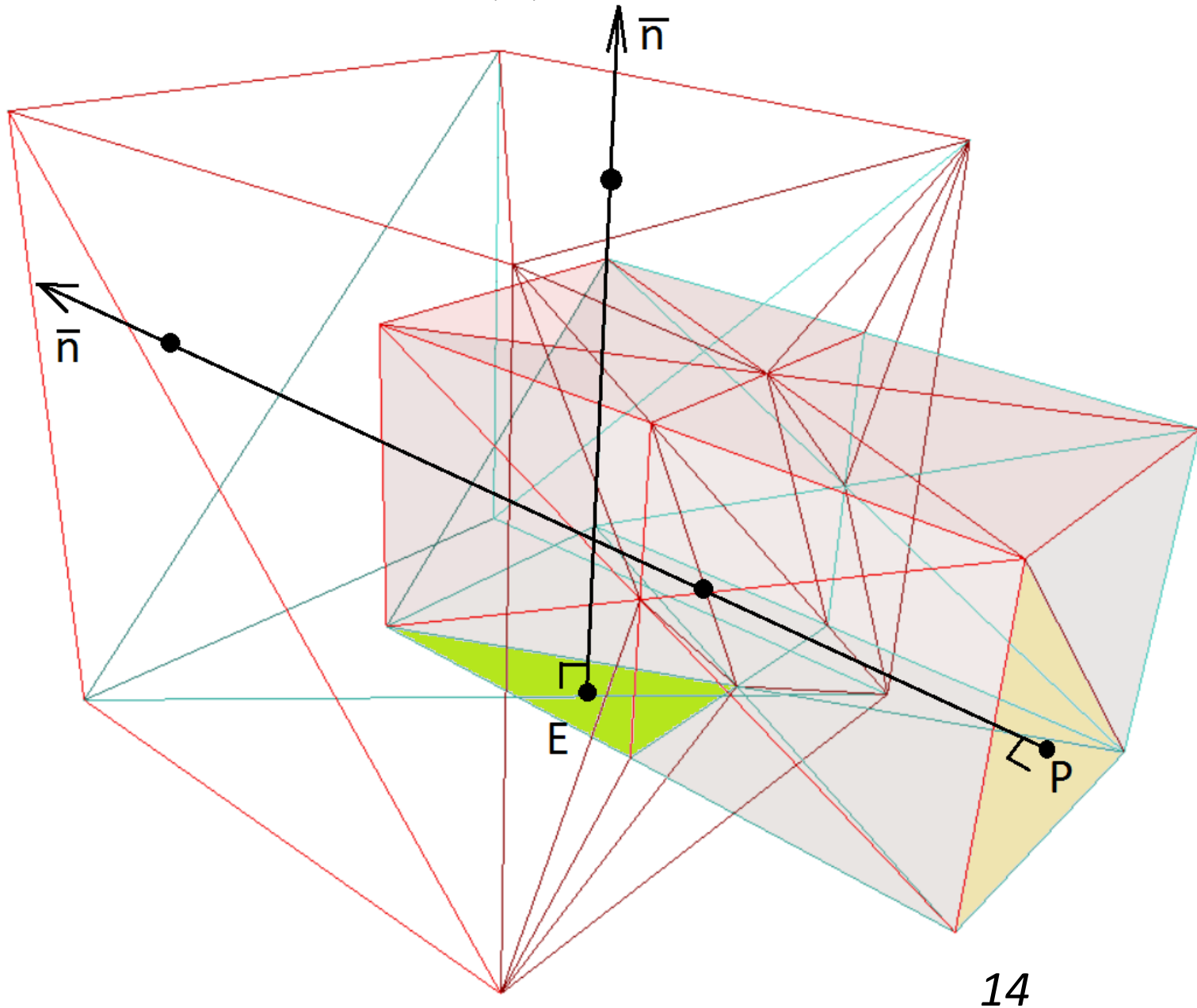


До



После

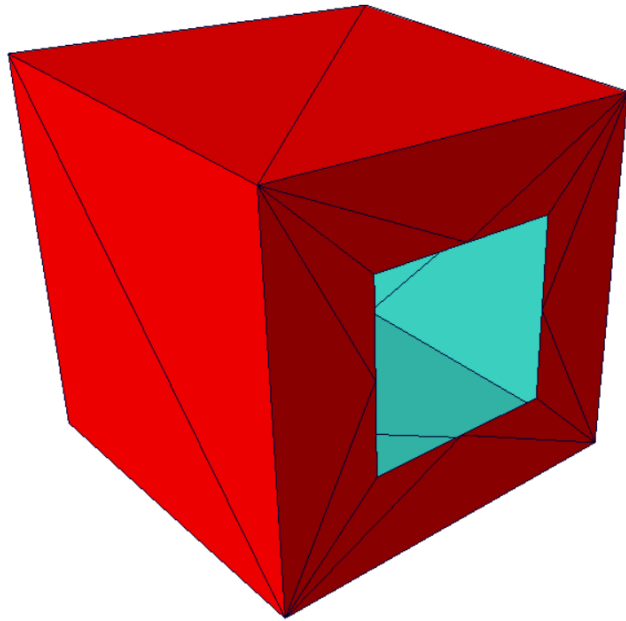
# Разбиение моделей на внешние и внутренние части



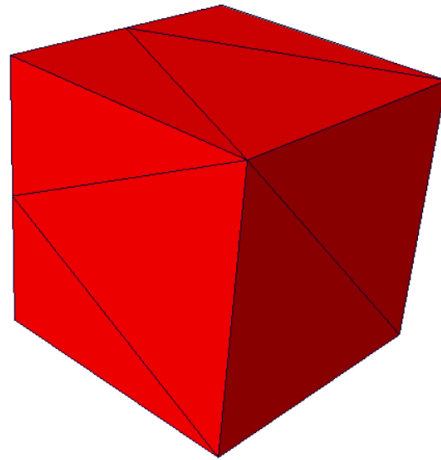
Чётное количество  
пересечений – внешняя часть,  
нечётное -внутренняя

# Получение результата операции путем сложения частей моделей

Внешние части моделей

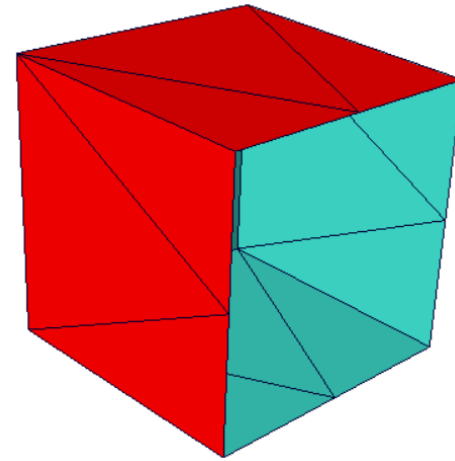


1

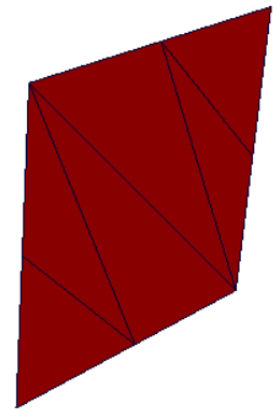


2

Внутренние части моделей



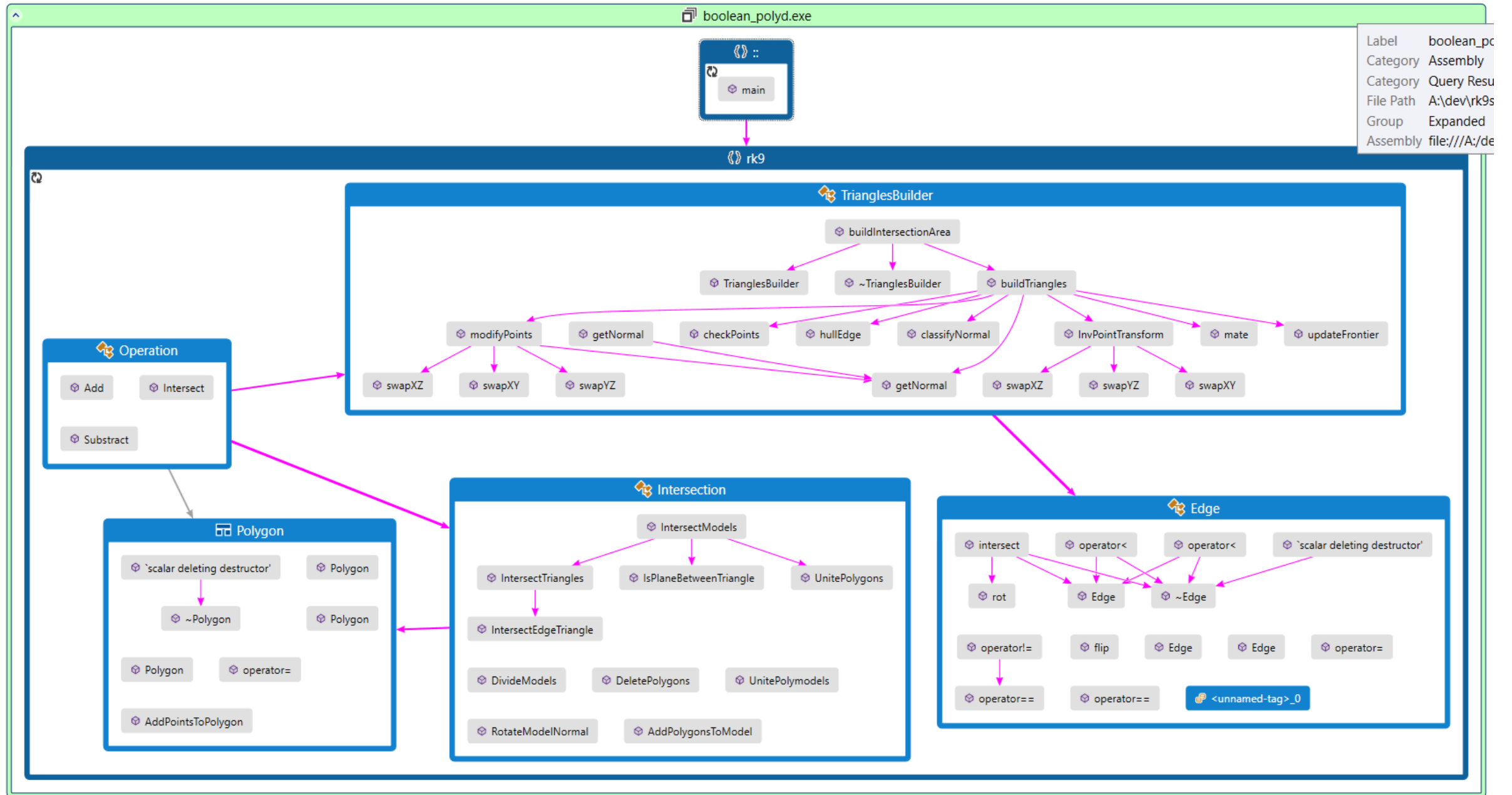
3



4

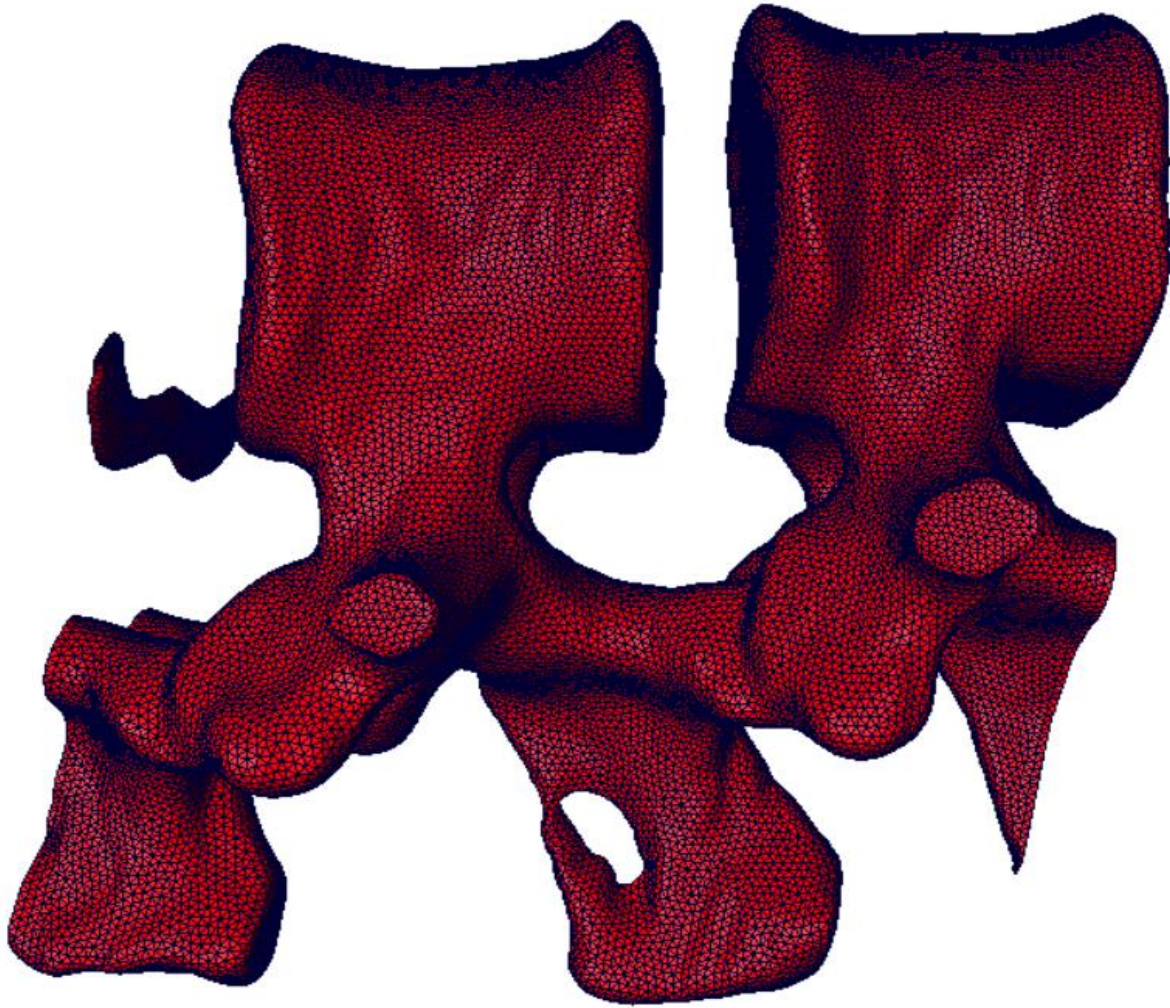
1. Объединение: 1 и 2
2. Пересечение: 3 и 4
3. Вычитание: 1 и 3 или 2 и 4

# Code Map

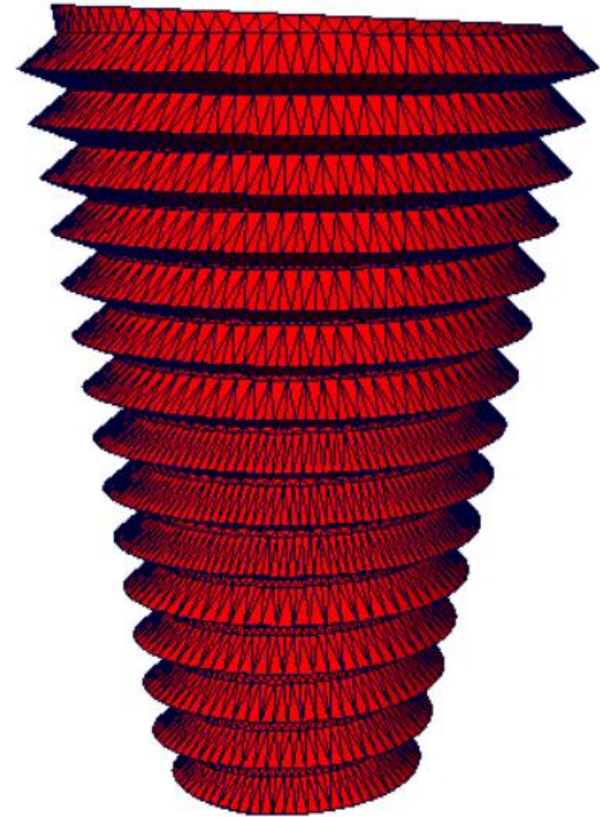




# Тестирование

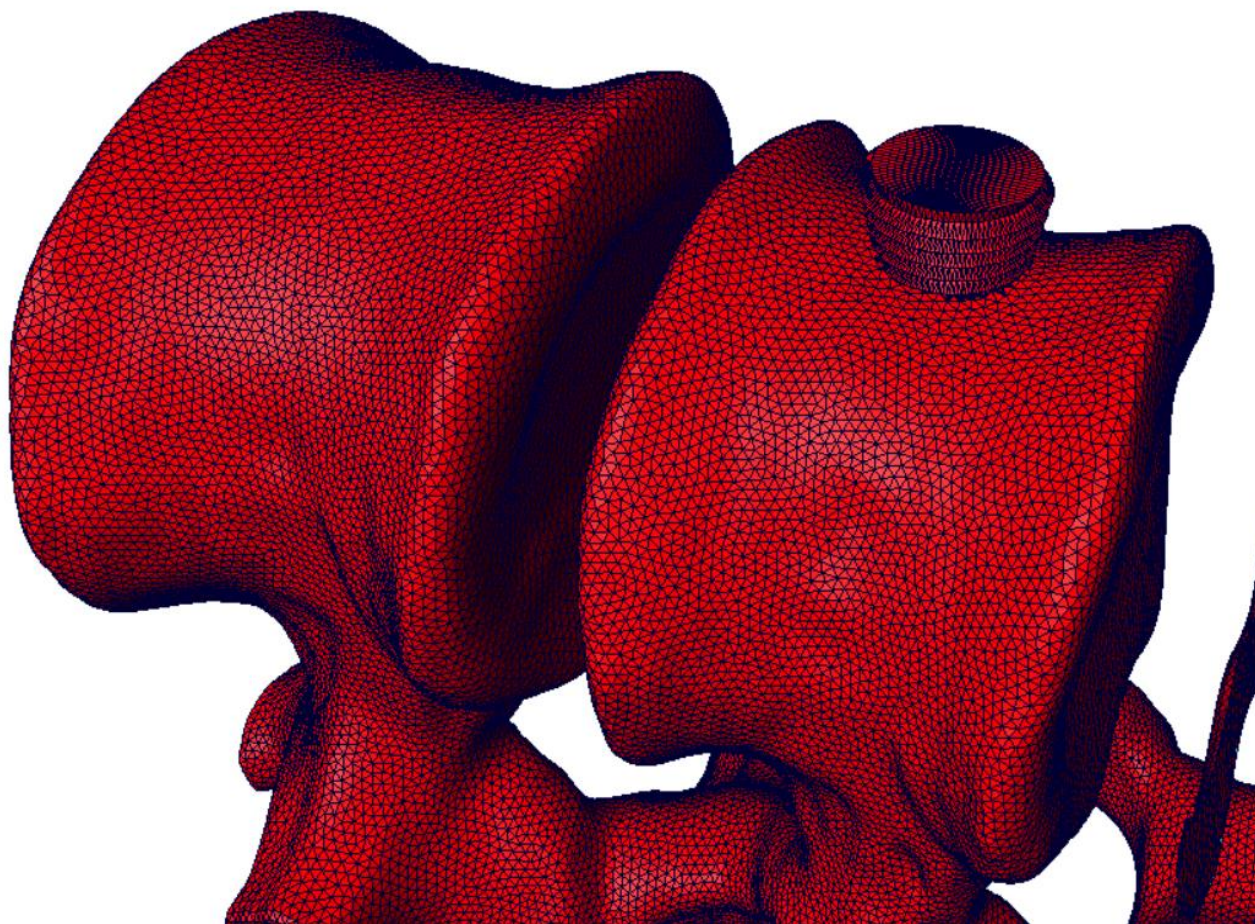


Позвонки

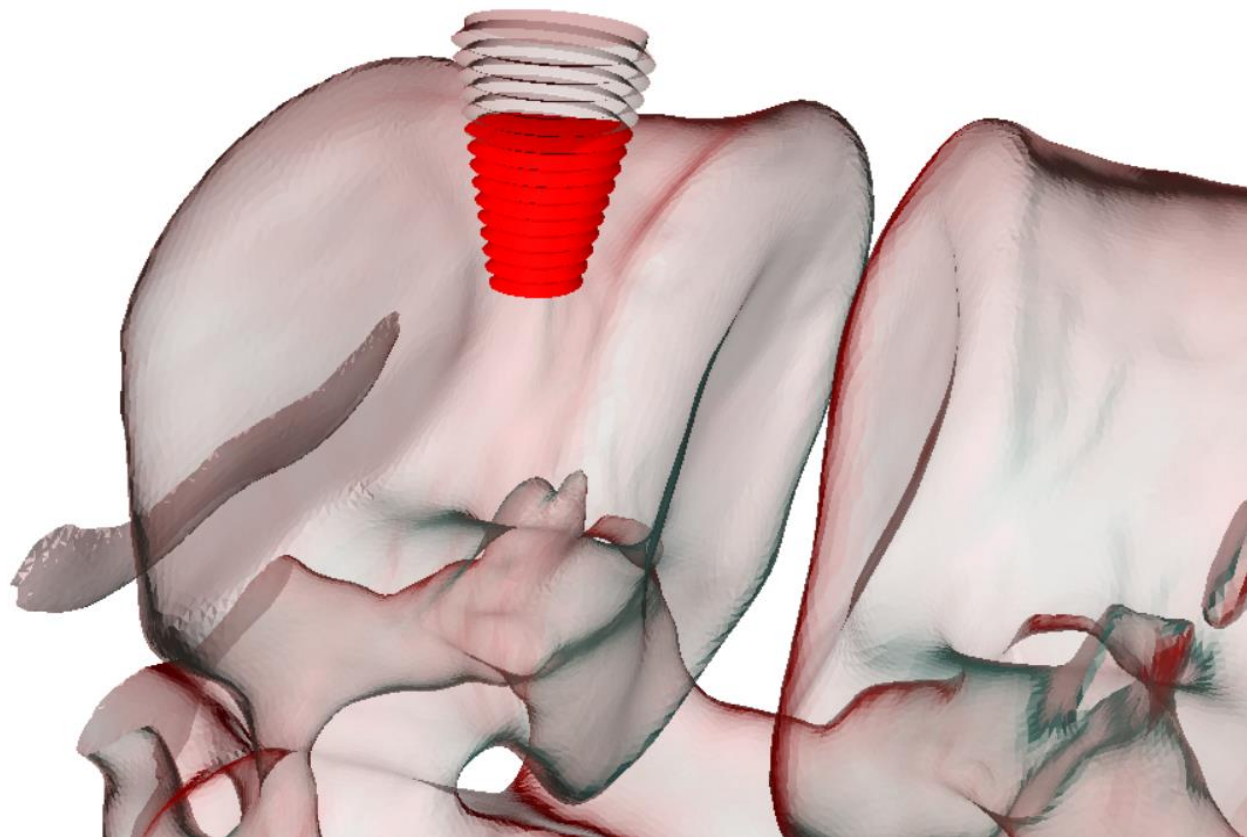


Конический винт





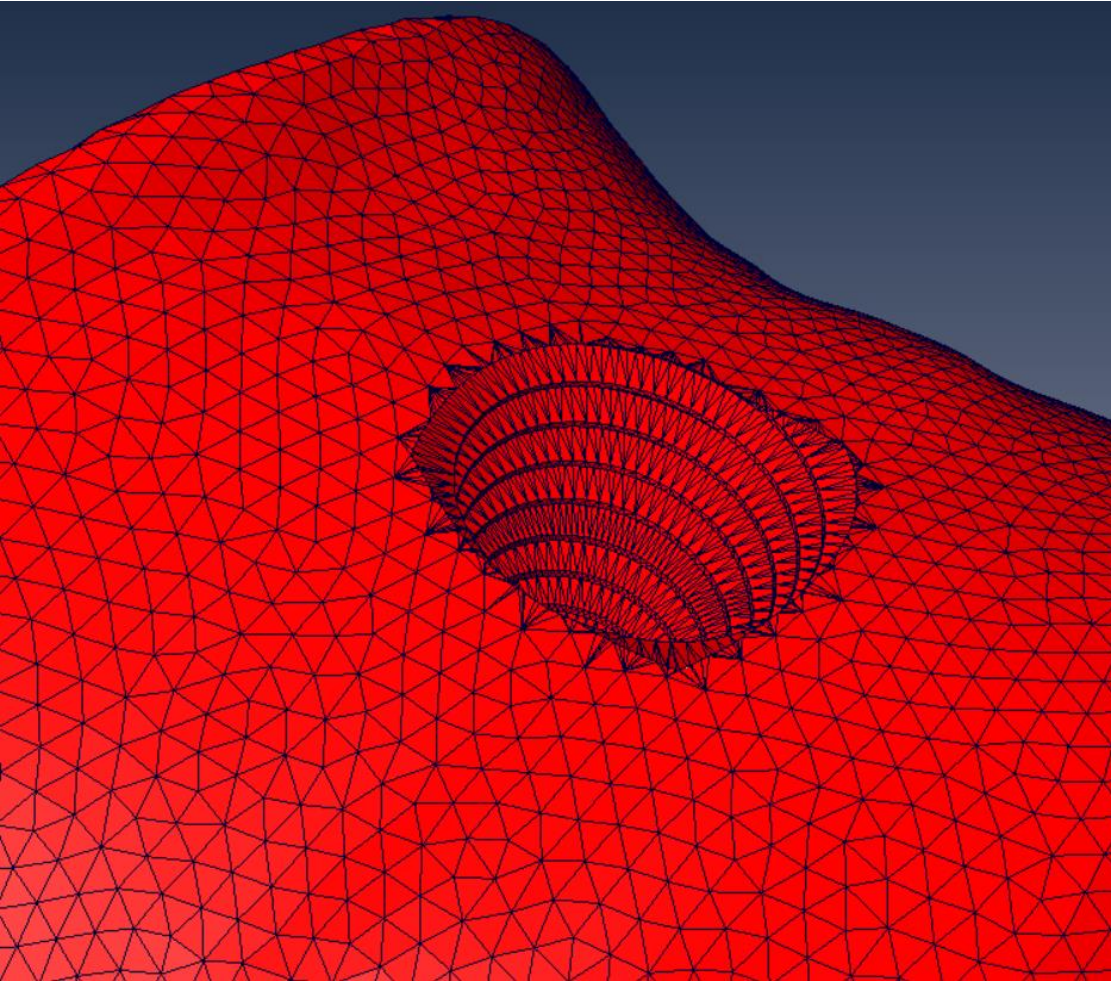
Объединение



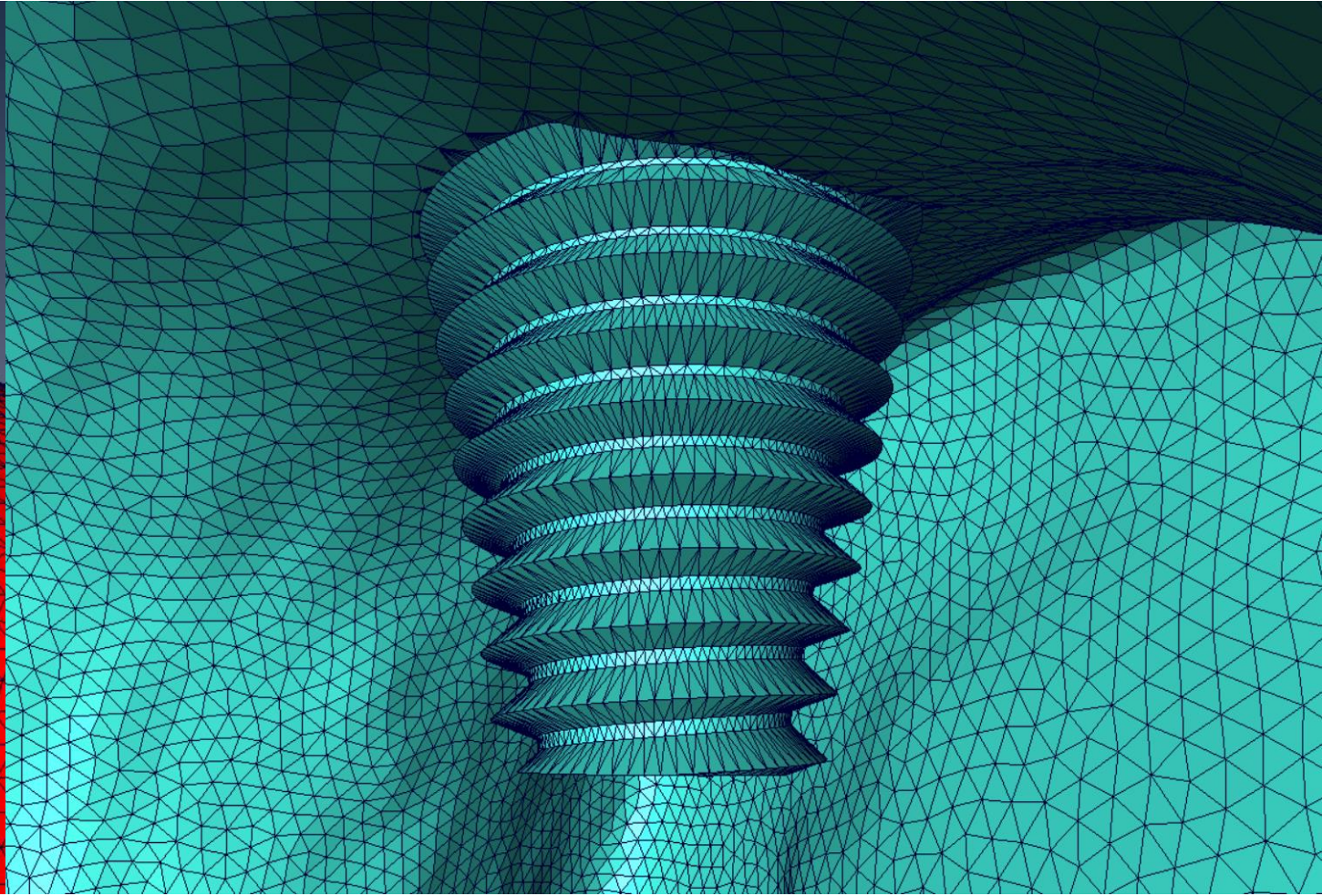
Пересечение



# Вычитание конического винта из позвонков



Вид снаружи



Вид изнутри



# Результаты тестирования

Тестируемые модели (количество треугольников)	Операция	Количество дыр	Non- manifold edges*	Время операции, с
Конус/сфера (378/1280)	Объединение	0	0	<1
	Пересечение	0	0	<1
	Вычитание	0	0	<1
Куб/параллелепипед (12/12)	Объединение	0	0	<1
	Пересечение	0	0	<1
	Вычитание	0	0	<1
Позвонки/ Конический винт (95986/9294)	Объединение	0	2	32
	Пересечение	1	0	32
	Вычитание	0	2	32

\*Non-manifold edges – это ребра, которые включены в более, чем два треугольника

# Дальнейшее развитие

- Улучшение быстродействия
- Интеграция кода, написанного однопользователями

Спасибо за внимание