|  |  |
| --- | --- |
|  | **Министерство науки и высшего образования Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение**  **высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

ФАКУЛЬТЕТ *Робототехники и комплексной автоматизации*

КАФЕДРА *Системы автоматизированного проектирования (РК-6)*

**ОТЧЕТ О ВЫПОЛНЕНИИ ДОМАШНЕГО ЗАДАНИЯ №1**

по курсу: «Автоматизация технологического проектирования»

|  |  |  |
| --- | --- | --- |
| Студент |  | Жидков Антон Алексеевич |
| Группа |  | РК6-31М |
| Тип задания |  | Домашнее задание |
|  |  |  |

Студент **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_Жидков А.А.**

*подпись, дата фамилия, и.о.*

Преподаватель **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_Божко А.Н.\_**

*подпись, дата фамилия, и.о.*

*Москва, 2024 г.*

Оглавление

[**1.** **Задание** 3](#_Toc185959139)

[**2.** **Введение** 4](#_Toc185959140)

[**3.** **Алгоритмы** 4](#_Toc185959141)

[**3.1.** **Триангуляция Делоне** 4](#_Toc185959142)

[**3.2.** **Алгоритм Боуэра-Ватсона** 6](#_Toc185959143)

[**3.3.** **Алгоритм Уайлера-Атертона** 8](#_Toc185959144)

[**3.4.** **Составление графа и поиск кратчайшего пути** 9](#_Toc185959145)

[**4.** **Реализация** 9](#_Toc185959146)

[**5.** **Результаты работы** 15](#_Toc185959147)

# **Задание**

1. Разработать алгоритм, осуществляющий триангуляцию свободного пространства и поиск кратчайшего пути в заданном двумерном пространстве препятствий.

# **Введение**

# **Алгоритмы**

* 1. **Триангуляция Делоне**

Триангуляцией называется планарное разбиение плоскости на плоские фигуры, из которых одна является внешней бесконечностью, а остальные – треугольниками. Будем рассматривать задачу построения триангуляции по заданному набору S двумерных точек. Эта задача состоит в соединении заданных точек из S прямыми отрезками так, чтобы никакие отрезки не пересекались. Решение этой задачи неоднозначно, поэтому возникает проблема построения оптимальной триангуляции. Оптимальной называют такую триангуляцию, у которой сумма длин всех ребер минимальна, однако построение такой триангуляции имеет сложность . Это ограничивает применение алгоритмов построения оптимальной триангуляции на практике.

Рассмотрим триангуляцию Делоне. Триангуляция Делоне – это такая триангуляция, при которой ни одна из точек набора S не попадает внутрь ни одной из описанных вокруг полученных треугольников окружностей за исключением точек, являющихся его вершинами.

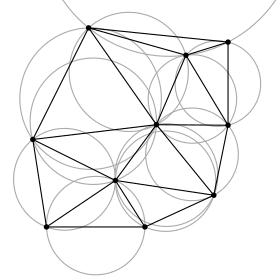


Рисунок 1 – пример триангуляции Делоне

Триангуляция Делоне не является оптимальной, но она строит набор треугольников, которые «стремятся к равноугольности» и имеет некоторые важные свойства: триангуляция Делоне обладает максимальной суммой минимальных углов всех своих треугольников среди всех возможных триангуляций на заданном наборе точек; триангуляция Делоне обладает минимальной суммой радиусов окружностей, описанных около треугольников, среди всех возможных триангуляций на заданном наборе точек.

Описание итеративного алгоритма Делоне.

Итеративный алгоритм предполагает пошаговое создание триангуляции Делоне, где на каждом шаге добавляется новый треугольник, соответствующий критерию Делоне, к уже существующей структуре. При добавлении точки могут возникнуть четыре основных ситуации:

* Точка находится внутри одного из существующих треугольников триангуляции.
* Точка расположена за пределами текущей триангуляции.
* Точка лежит на ребре одного из треугольников.
* Точка совпадает с одной из вершин треугольников триангуляции.

В случае, если точка попадает внутрь треугольника, последний разделяется на несколько новых, с обязательной проверкой условия Делоне. Если точка находится за пределами триангуляции, добавляются новые внешние треугольники с последующей проверкой на соответствие условию. Когда точка оказывается на ребре, происходит разделение ребра и связанных с ним треугольников. Если же точка совпадает с одной из существующих вершин, её добавление игнорируется.

Общая сложность алгоритма складывается из этапов поиска подходящего треугольника для добавления точки, создания новых треугольников, проверки выполнения условия Делоне и перестроения триангуляции при необходимости. Поиск треугольника оценивается как , тогда как построение новых треугольников и проверка их соответствия обычно требуют фиксированного числа операций.

Ситуацию, когда точка находится вне триангуляции, можно избежать, если начать процесс с большого стартового треугольника, охватывающего все точки. При добавлении каждой новой точки может потребоваться перестроение значительной части триангуляции, что приводит к общей вычислительной сложности алгоритма порядка .

* 1. **Алгоритм Боуэра-Ватсона**

Алгоритм Боуэра-Ватсона – инкрементный алгоритм, который позволяет получить триангуляцию Делоне для конечного набора точек. Для создания триангуляции Делоне задаётся супер-треугольник, который охватывает все заданные точки. Далее точки добавляются к действительной триангуляции Делоне по одной. На рисунке 1 изображен супер-треугольник с первой добавленной точкой.

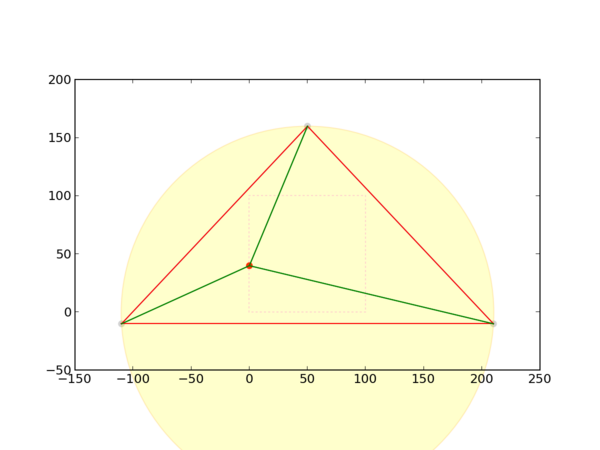


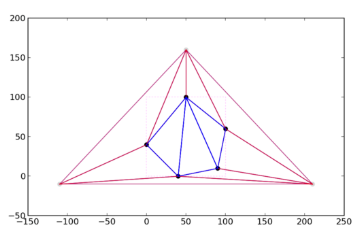
Рисунок 2 – Вставка точки в супер-треугольник

После добавление точки происходит проверка условия непринадлежности точки к описанной окружности каждого треугольника триангуляции. На рисунке 2 видно, что описанная вокруг супер-треугольника окружность содержит добавленную точку. В таком случае нужно перестроить триангуляцию. Для этого все треугольники, описанные окружности которых содержат точку, образуют границы многоугольной пустоты, после чего данные треугольники удаляются из триангуляции. Затем происходит процесс триангуляции многоугольной пустоты и добавление её к исходной триангуляции. Результат данного этапа представлен на рисунке 3.



Рисунок 3 – Результат перестроения триангуляции и добавление ещё одной точки.

На рисунке 3 визуально разделены окружности, описанные вокруг треугольников, содержащие точку, не являющейся вершиной треугольника, (желтым цветом) и не содержащие подобных точек (серые). После добавления всех точек получим триангуляцию внутри супер-треугольника (рисунок 4). После удаления супер-треугольника будет найдена искомая триангуляция Делоне.

  
Рисунок 4 – Триангуляция внутри супер-треугольника.

## **Алгоритм Уайлера-Атертона**

Алгоритм Уайлера-Атертона позволяет найти область пересечения отсекаемого многоугольника по отсекающему многоугольнику. Отсекаемый и отсекающий многоугольники могут быть невыпуклыми. Алгоритм является итерационным и описывается как:

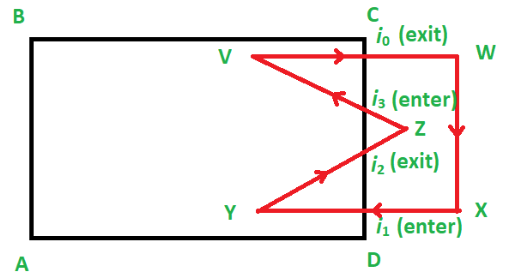
1. Составление списка из координат вершин A и B с сортировкой вершин по часовой стрелке для полигона А и В.
2. Определение точек пересечения А и В при проходе полигона по часовой стрелке с пометкой о том, входит ли полигон В в полигон А в этой точке или выходит.
3. 

Рисунок 5 – Определение точек пересечения при обходе полигона VWXY по часовой стрелке

1. Создание и заполнение двух списков обхода полигонов А и В с добавлением точек пересечения многоугольников.

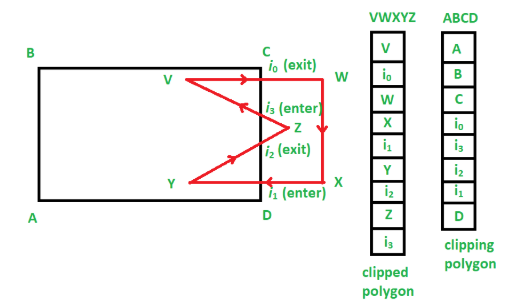
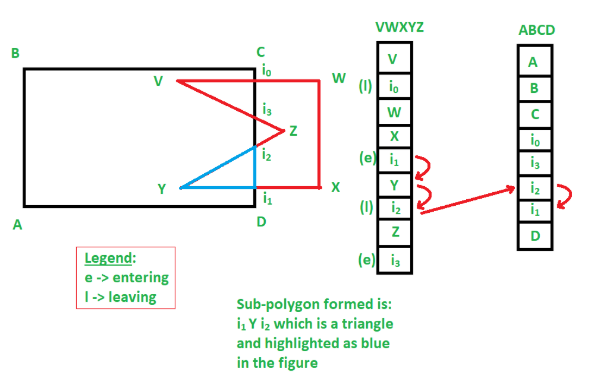


Рисунок 6 – Составление списков обхода полигонов

1. Обход полигона по первому списку, поиск первой входящей точке пересечения. При нахождении входящей точки пересечения, обход продолжается, начиная с этой точки во втором списке. При нахождении следующей точки пересечения образуется замкнутый контур из рассмотренных точек, который является один из наложений препятствий.

  
Рисунок 7 - Нахождение контура наложения препятствий.

1. После нахождения всех контуров наложения полигонов производится операция отсечения полученных контуров из первого полигона.
2. Если ни одного пересечения не найдено, возникает одна из следующих ситуаций:
   1. A внутри B — вернуть A при отсечении, B при объединении.
   2. B внутри A — вернуть B при отсечении, A при объединении.
   3. A и B не пересекаются — вернуть пустое множество при отсечении, A&B при объединении.

## **Составление графа и поиск кратчайшего пути**

Для составления графа разработан алгоритм прохождения по всем треугольникам полученной триангуляции и добавления характерных точек треугольника в итоговый граф. В качестве характерных точек выбраны центр окружности, описанной вокруг треугольника и середины сторон треугольника. Перед каждым добавлением точки в граф производится проверка принадлежности этой точки к препятствию.

При триангуляции поля с многогранными препятствиями могут возникать случаи, когда рассмотренные характерные точки не обеспечивают составление графа. Данные случаи встречаются при такой триангуляции, когда в треугольник попадает часть препятствия. В этом случае предлагается добавить в граф середины отрезков, не лежащих на препятствии в случае, если середина стороны треугольника принадлежит препятствию. Для этого в алгоритм добавлен поиск точек пересечения препятствий и добавления середин соответствующих отрезков.

Рёбра графа соответствуют связям добавляемых вершин внутри треугольника по принципу «все со всеми». Перед добавлением связи выполняется проверка на достижимость из одной точки в другую, то есть не пересекает ли путь препятствие.

# **Реализация**

Для построения заданного пространства было реализовано считывание начальной и целевой точек и препятствий из генерируемого файла формата .json.

def filter\_by\_type(data, type\_name):

    return [item for item in data if item.get('type') == type\_name]

info\_elements = filter\_by\_type(data, 'info')

start\_points = filter\_by\_type(data, 'startPoint')

end\_points = filter\_by\_type(data, 'endPoint')

polygons = filter\_by\_type(data, 'polygon')

Был описан ряд сущностей для реализации алгоритма: класс точки и препятствия.

# **Результаты работы**

В результате работы алгоритма были получены графики: исходного пространства, триангуляции свободного пространства, итогового графа с кратчайшим путём для препятствий разной плотности.

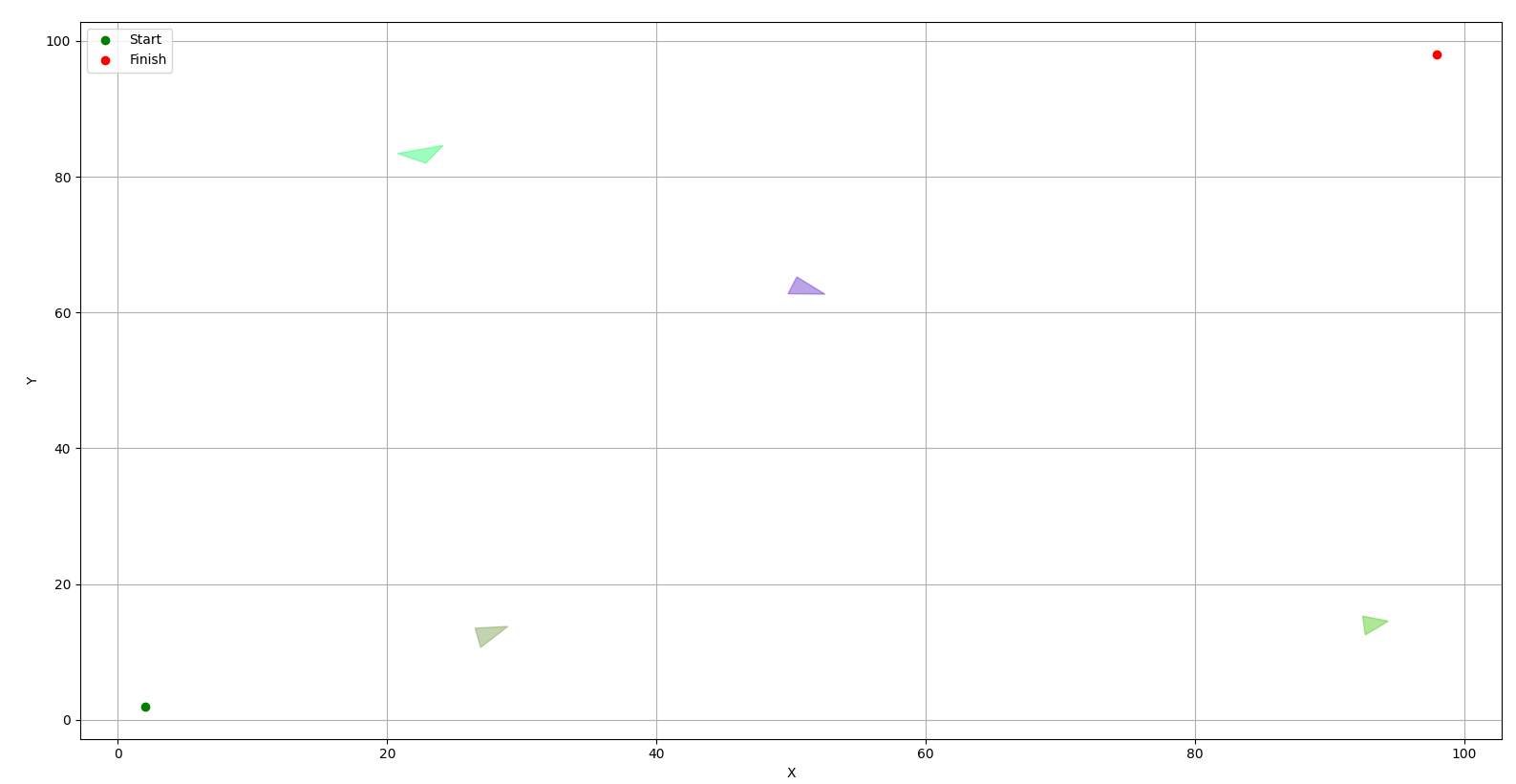


Рисунок 1 - Исходное пространство.

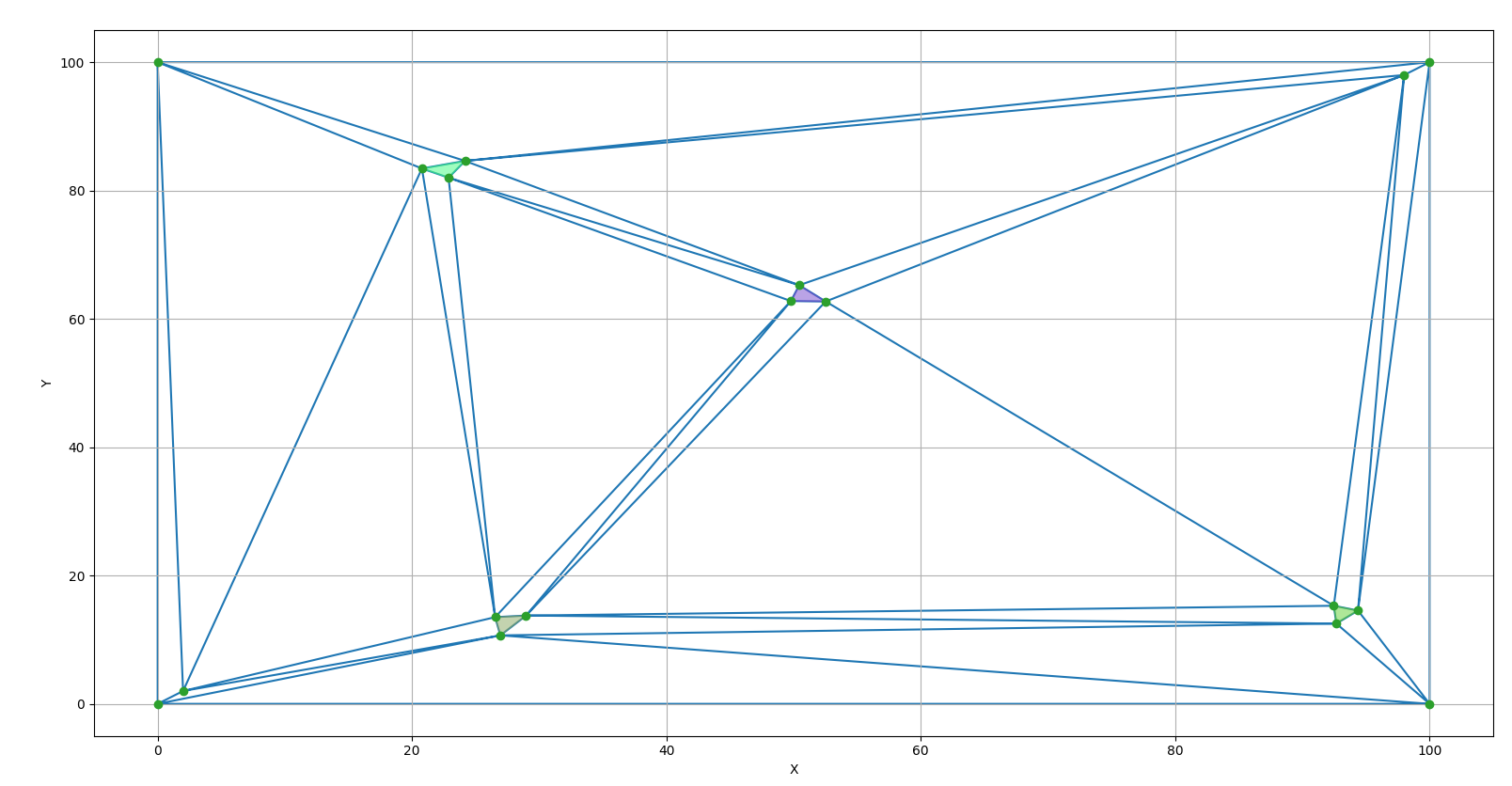


Рисунок 2 – Триангуляция свободного пространства.

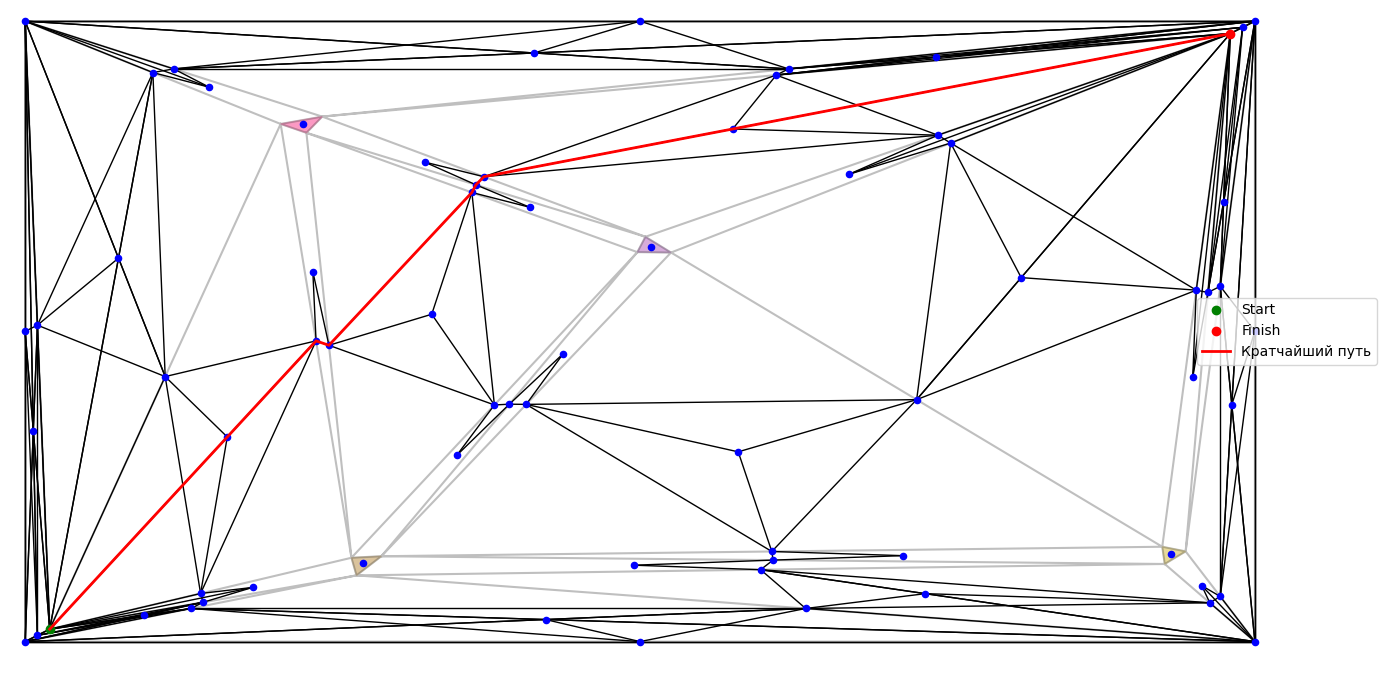


Рисунок 3 – Итоговый граф и кратчайший путь.

Для пространства со средней плотностью препятствий:

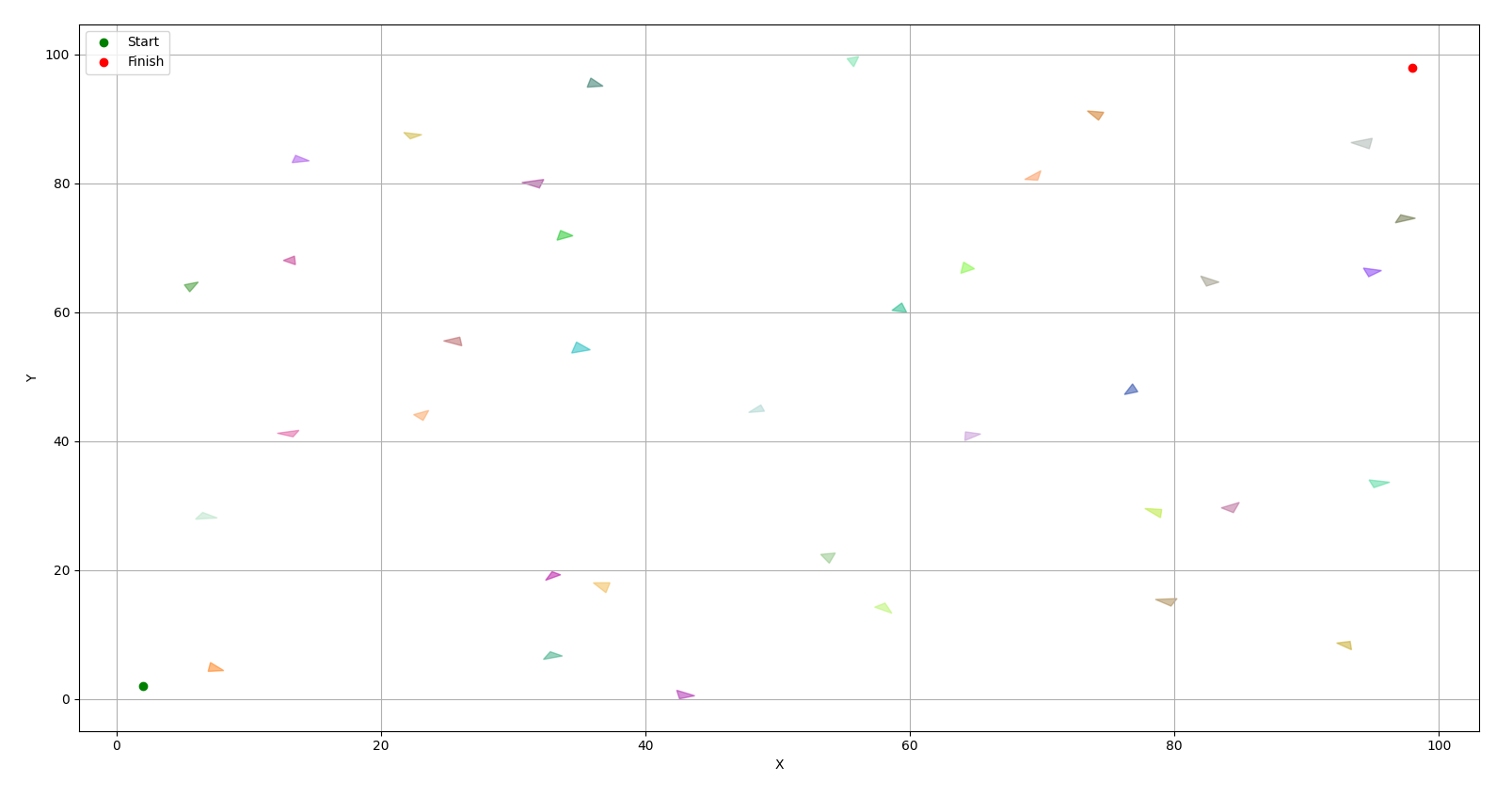


Рисунок 4 – Исходное пространство.

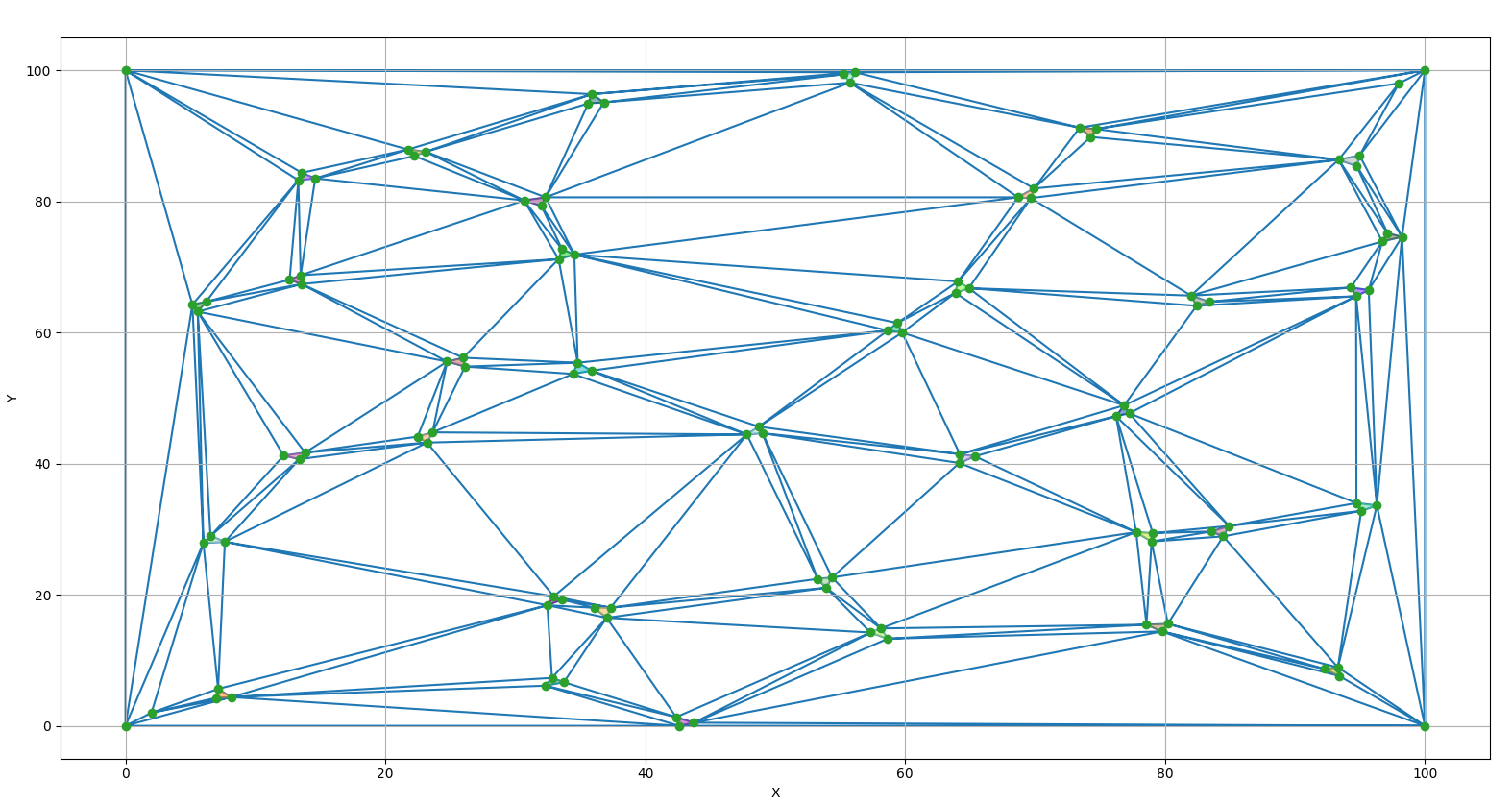


Рисунок 5 – Триангуляция свободного пространства

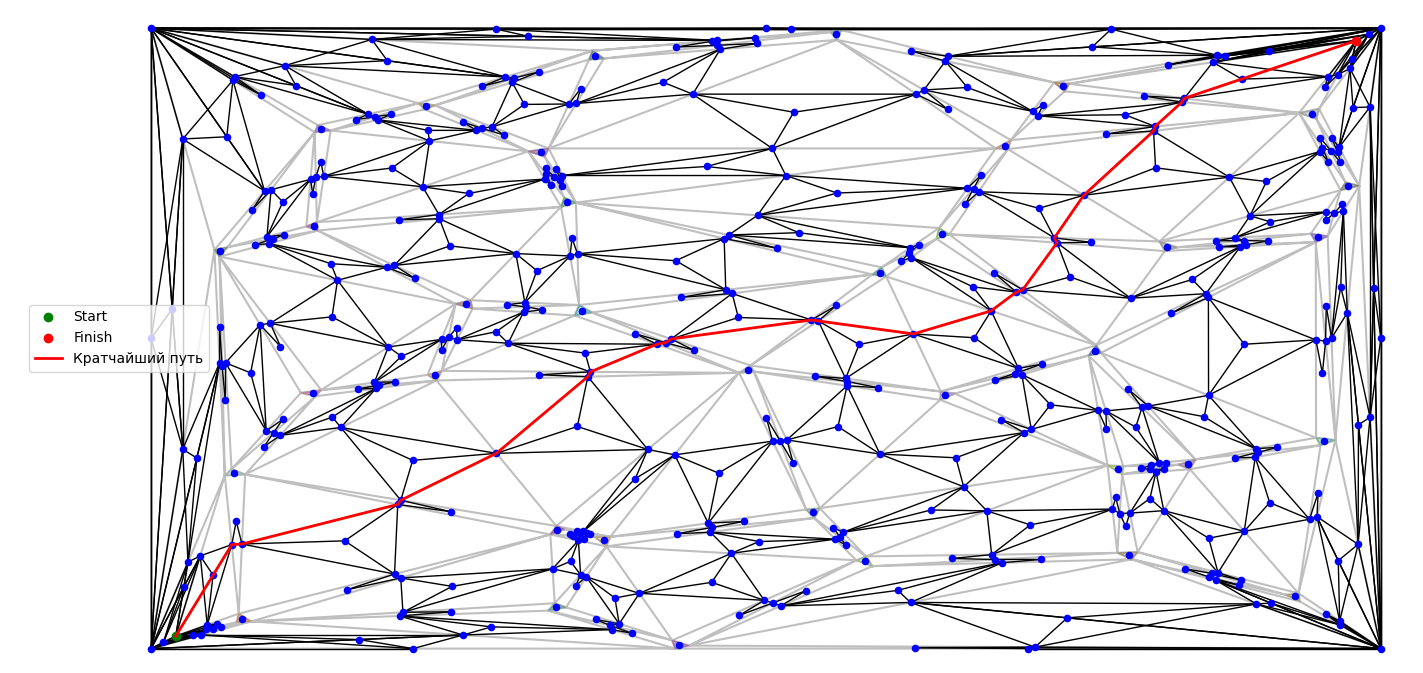


Рисунок 6 – Итоговый граф и кратчайший путь.

Для пространства с высокой плотностью препятствий:

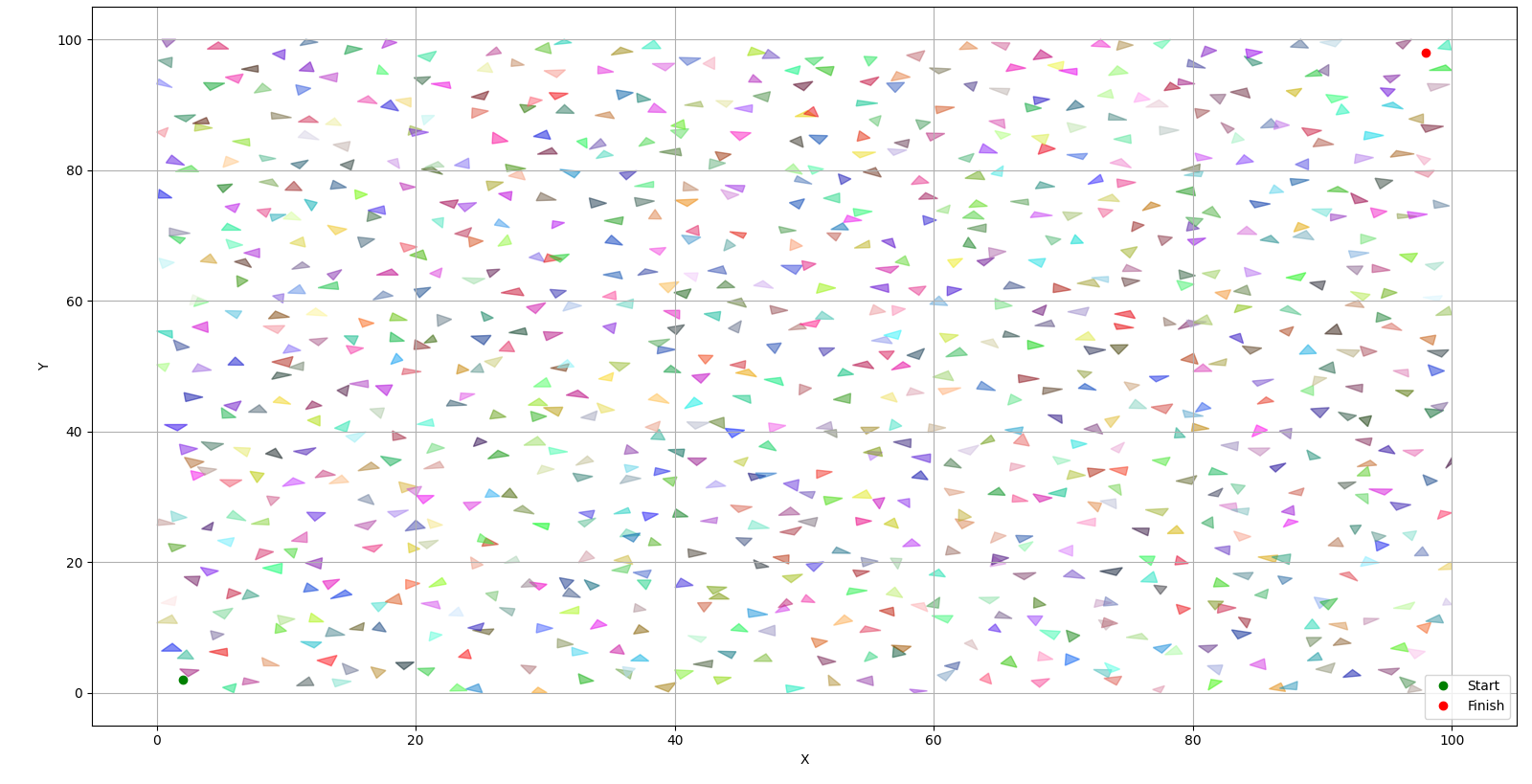


Рисунок 7 – Исходное пространство.

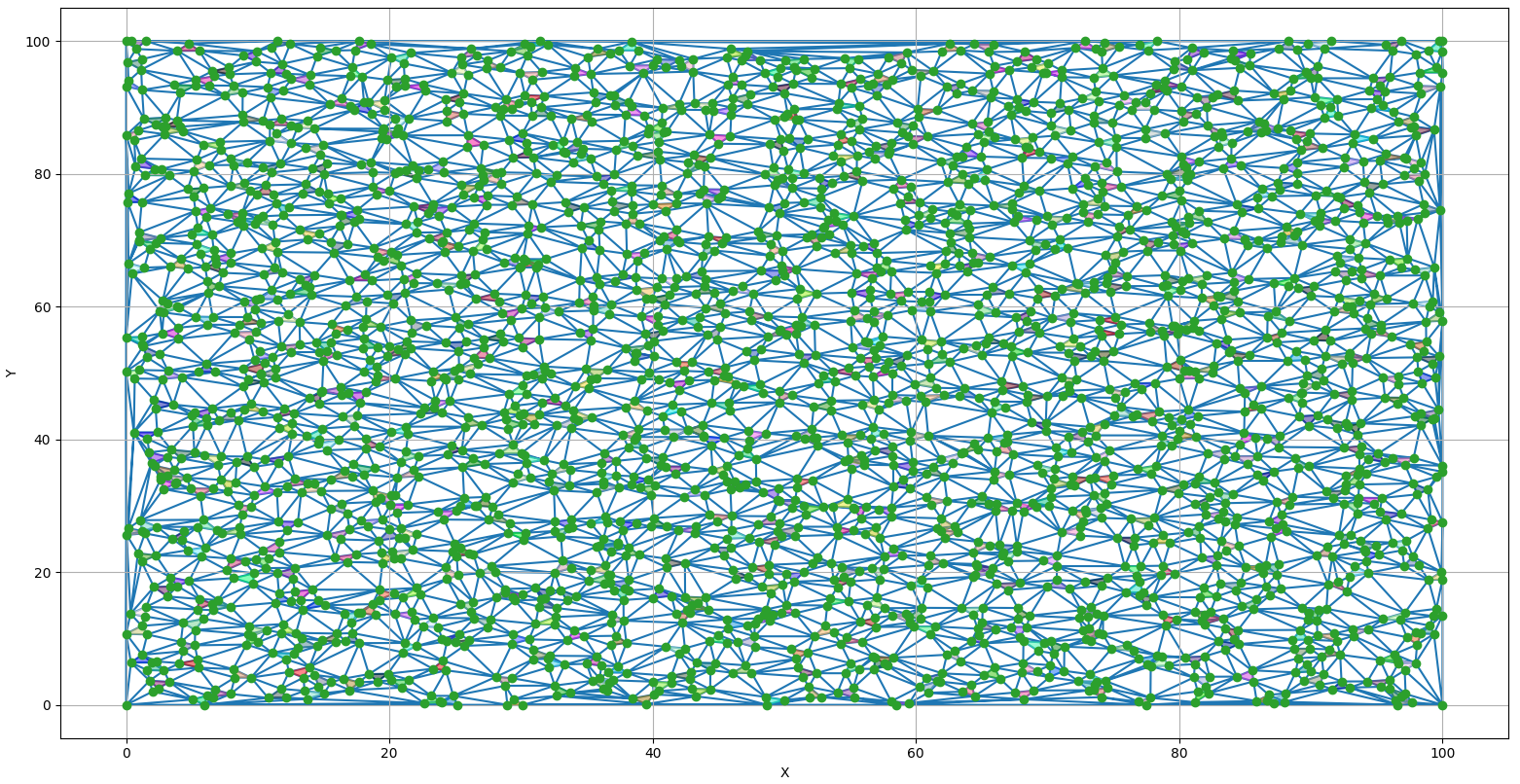


Рисунок 8 – Триангуляция свободного пространства.

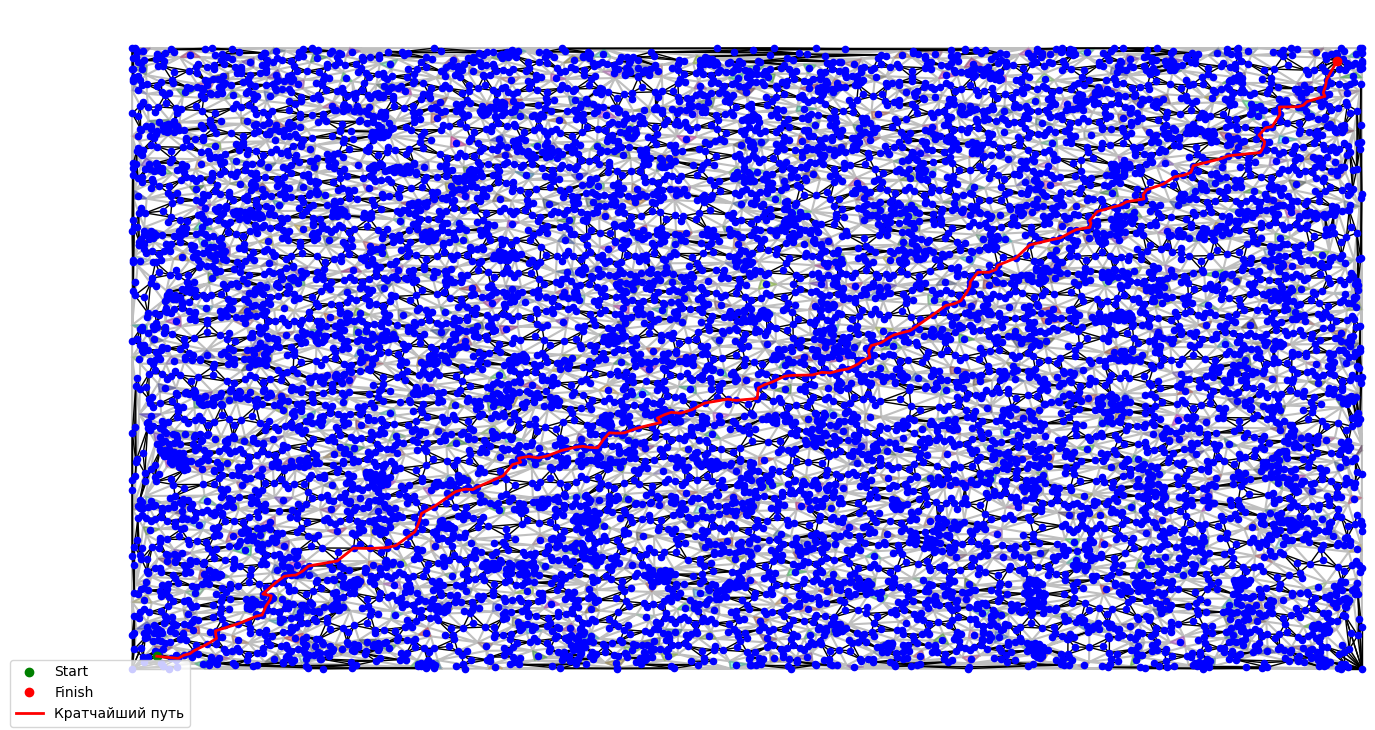


Рисунок 9 – Итоговый граф и кратчайший путь

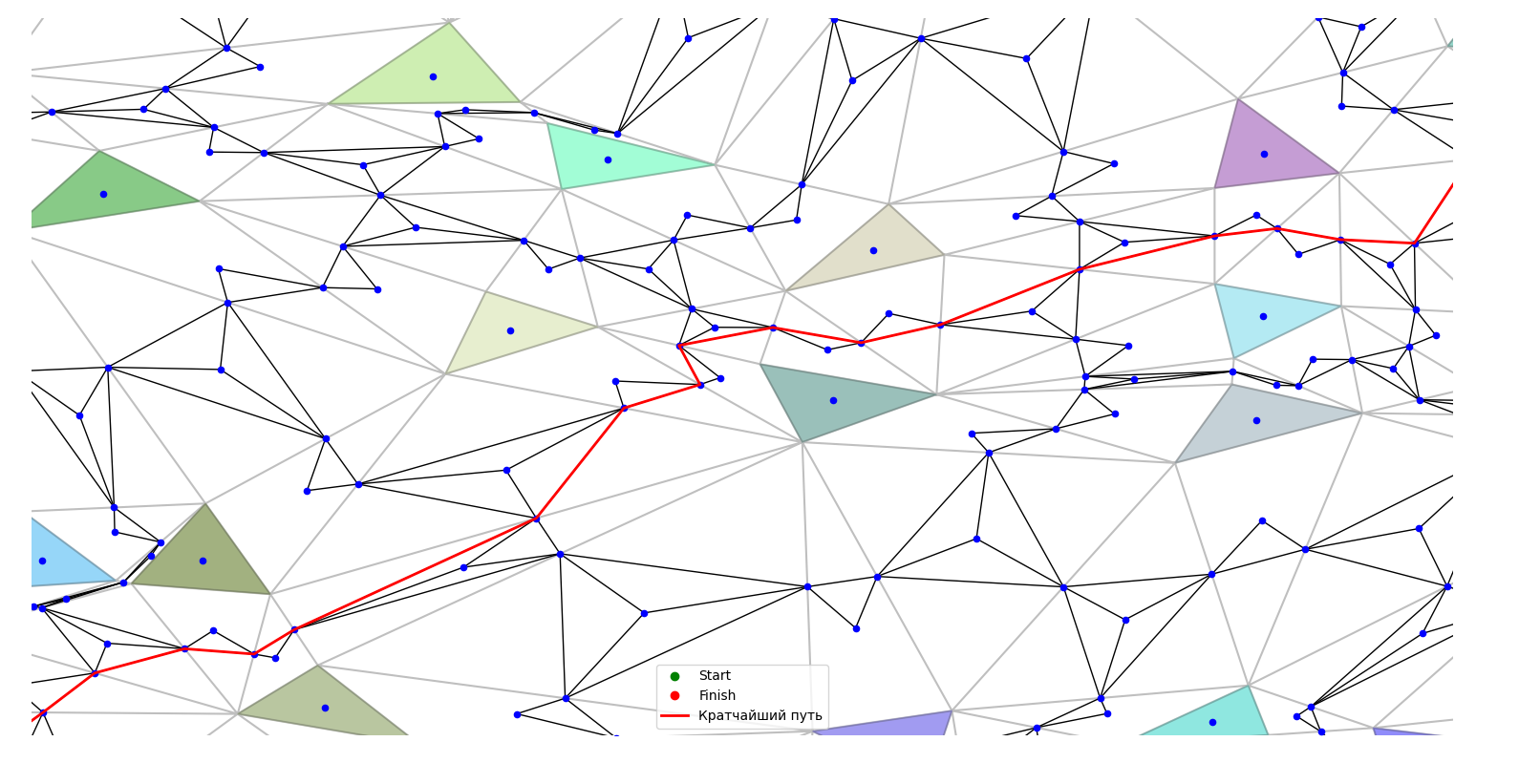


Рисунок 10 – Итоговый граф и кратчайший путь

Итоговое время выполнения:

|  |  |
| --- | --- |
| Плотность препятствий | Время выполнения |
| 3 | 0.025010108947753906 |
| 10 | 3.837907075881958 |
| 30 | 309.3849790096283 |
| 40 | 96.65564632415771 |

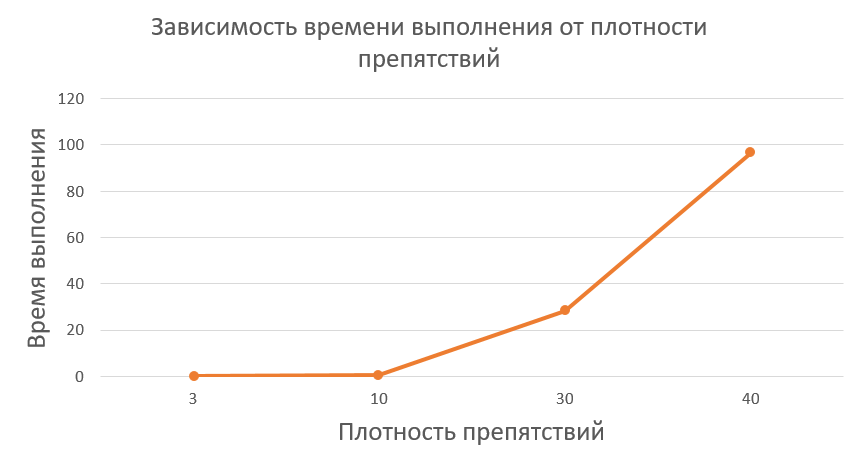


Рисунок 11 – Зависимость времени выполнения от плотности препятствий.