Артюшина А.М. Построение маршрута при помощи точной декомпозиции. программная реализация метода продолжения границ // Международный журнал информационных технологий и энергоэффективности. – 2022. – Т. 7 № 4(26) часть 2 с. 34–42

Международный журнал информационных технологий и энергоэффективности

Сайт журнала:
http://www.openaccessscience.ru/index.php/ijcse/

УДК 004.4

ПОСТРОЕНИЕ МАРШРУТА ПРИ ПОМОЩИ ТОЧНОЙ ДЕКОМПОЗИЦИИ. ПРОГРАММНАЯ РЕАЛИЗАЦИЯ МЕТОДА ПРОДОЛЖЕНИЯ ГРАНИЦ

Артюшина А.М.

Московский государственный технический университет имени Н.Э. Баумана, Москва, Россия (105005, г. Москва, улица 2-я Бауманская, д. 5, к. 1), e-mail: artyushina_n@mail.ru

Рассмотрено применение методов точной декомпозиции для решения задачи планирования перемещений. Разработана программная реализация поиска маршрута в конфигурационном пространстве методом продолжения границ препятствий. Поиск пути в канальном графе реализован с помощью алгоритма Ли. Определена зависимость времени выполнения программы от количества вершин препятствий, создаваемых с помощью программы-генератора. В процессе работы был использован язык программирования Python.

Ключевые слова: планирование перемещений, точная декомпозиция, конфигурационное пространство, метод продолжения границ препятствий.

MOTION PLANNING WITH EXACT DECOMPOSITION.SOFTWARE IMPLEMENTATION OF THE BOUNDARY EXTENSION METHOD

Artyushina A.M.

Bauman Moscow State Technical University, Moscow, Russia (105005, Moscow, 2nd Baumanskaya street, 5, building 1); e-mail: artyushina_n@mail.ru

The application of methods of exact decomposition to the motion planning problem is considered. A software implementation of the Obstacle Boundary Continuation Method for pathfinding in configuration space has been developed. The search for a path in a channel graph is implemented using the Lee algorithm. The dependence of the program execution time on the number of obstacle vertices created using the generator program is determined. The implementation of the program is written in Python.

Keywords: motion planning; exact decomposition; configuration space; obstacle boundary extension method.

Введение

Построение маршрута движения является классической задачей искусственного интеллекта и одной из важнейших задач в навигации роботов. Обычно под планированием движения понимается поиск бесконфликтного пути для перемещения твердого тела в пространственно-трехмерной сцене. Искомый путь представляет собой непрерывную кривую в конфигурационном пространстве объекта, которая соединяет его начальное и конечное положения, исключает столкновения с препятствиями сцены и удовлетворяет всем установленным ограничениям.

Описание алгоритма точной декомпозиции методом продолжения границ

Наиболее простой и распространенный подход к планированию движения состоит в применении методов пространственной декомпозиции. Данные методы предполагают дискретизацию среды с ограничениями на множество простых областей [1], [2]. Если говорить в общем, планирование пути с помощью алгоритмов точной декомпозиции представляет собой нахождение последовательности подобластей, образующих связный граф, которые должен пройти робот, чтобы попасть из начальной точки в конечную точку. При решении задачи нахождении маршрута с помощью пространственной декомпозиции, можно использовать разные методы нахождения пути в графе.

Существуют методы регулярной и объектно-зависимой декомпозиции. Первый класс методов предполагает разбиение всего объема конфигурационного пространства сеткой с фиксированным размером ячеек. Методы объектно-зависимой или точной декомпозиции дискретизируют пространство на области, полностью свободные от ограничений. Наиболее известными методами данного класса являются метод трапецеидальной декомпозиции, триангулированное разбиение, цилиндрическая декомпозиция, а также декомпозиция Морса. К данному типу методов также относится метод продолжения границ препятствий.

Поиск решения реализуется по следующему алгоритму.

1. Границы многоугольных препятствий продолжаются вплоть до столкновения с другими препятствиями или границами конфигурационного пространства.
2. Находятся середины полученных отрезков. Найденные точки являются вершинами канального графа.
3. Вершины соединяются линиями, не пересекающими препятствия, для получения связного канального графа.
4. К полученному графу ребрами присоединяются точки, описывающие начальное и конечное положения робота.
5. Реализуется поиск пути в графе.

Программная реализация

Программная реализация поиска маршрута в конфигурационном пространстве методом продолжения границ препятствий, была написана на языке программирования Python. Количество строк кода без поэтапной отрисовки результата: 340. Количество строк кода с поэтапной отрисовкой результата: 420. Полный листинг программы доступен по ссылке: https://github.com/StasyArt/Obstacle-boundary-extension-method-for-Motion-planning.

В программе были использованы следующие библиотечные модули и фреймворки:

- json: для взаимодействия с файлом препятствий, полученным в результате работы программы-генератора;
- math: для выполнения математических преобразований;
- copy: для глубокого копирования объектов;
- networkx: для работы с графами;
- numpy: для работы с многомерными массивами и матрицами;
- matplotlib.pyplot: для визуализации данных;
- matplotlib.lines: для работы с геометрическим примитивом прямой;

- geopandas: для визуализации пространственных данных;
- Polygon, LineString, Point from shapely.geometry: для работы с геометрическими фигурами;
- Line, Point from sympy: для работы с геометрическими фигурами;
- PolygonPatch: для последовательной визуализации результатов.

Рассмотрим процесс работы программы на конкретном примере. Предположим, что в результате работы программы-генератора было сформировано конфигурационное пространство с препятствиями, представленное на Рисунке 1.

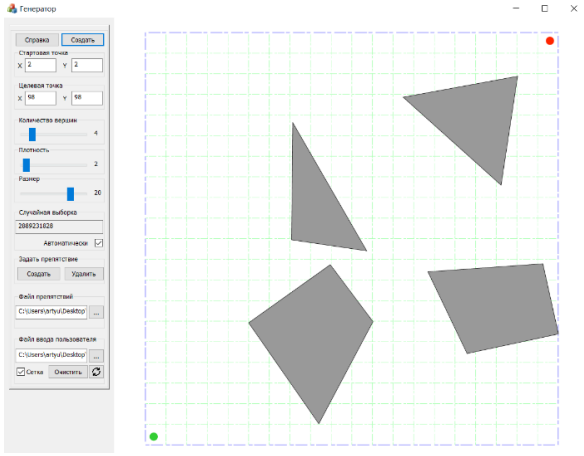

Рисунок 1 – Результат работы генератора препятствий

Файл 'ATP_generator/output_file.json', содержащий информацию о препятствиях считывается программой. В переменные startPoint и endPoint сохраняются координаты начальной и конечной точек положения робота. В массив polygon_arr сохраняются координаты вершин полигонов (препятствий). Результат считывания файла препятствий разработанной программой представлен на Рисунке 2.

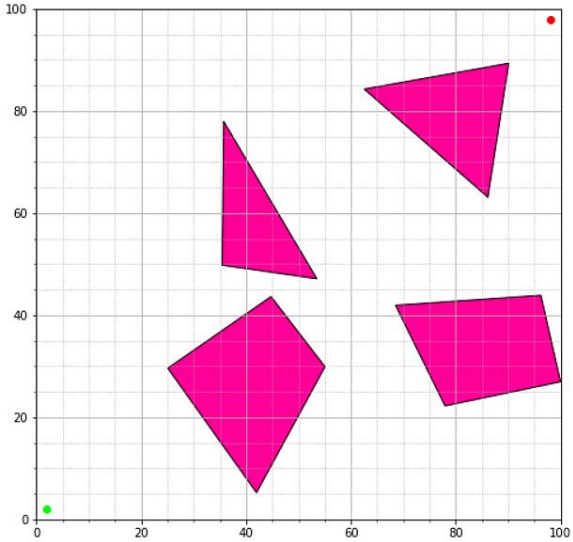

Рисунок 2 – Считывание конфигурационного пространства

Далее в программе строится продолжение сторон каждого препятствия вплоть до столкновения с границами конфигурационного пространства. Для этого, используя математическое уравнение прямой, находятся координаты точек пересечения продолжения

стороны полигона с продолжением границ пространства. Из четырех возможных точек выбираются те, которые пересекают прямые $x = 0,\ x = 100,\ y = 0,\ y = 100$ на границах конфигурационного пространства. Результат работы данного этапа представлен на Рисунке 3.

На следующем этапе (Рисунок 4) реализуется проверка пересечения полученных линий с другими препятствиями. Находятся координаты точек пересечения, и каждая линия ограничивается ближайшим препятствием, с которым она сталкивается.

Далее в программе с помощью математической формулы, использующей известные координаты концов каждого полученного отрезка, находится середина каждого отрезка и отображается в конфигурационном пространстве (Рисунок 5). Найденные точки являются вершинами канального графа (Рисунок 6).

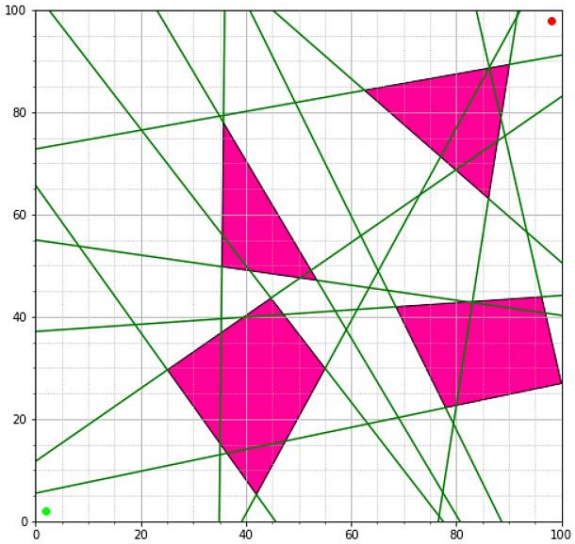

Рисунок 3 – Продолжение границ препятствий до пересечения с границами конфигурационного пространства

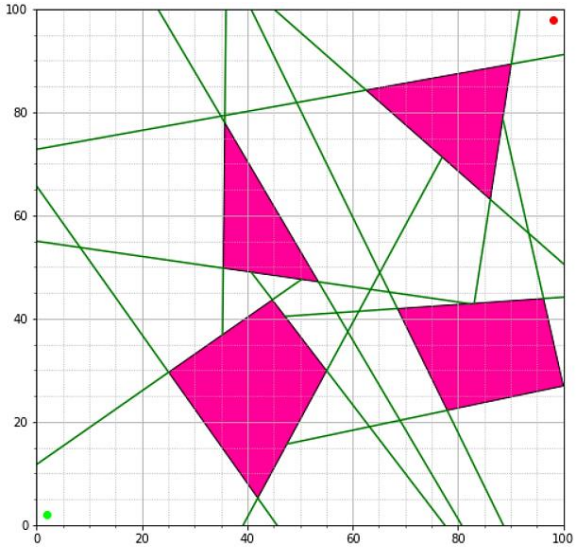

Рисунок 4 – Продолжение границ препятствий до пересечения с границами конф. пространства и другими препятствиями

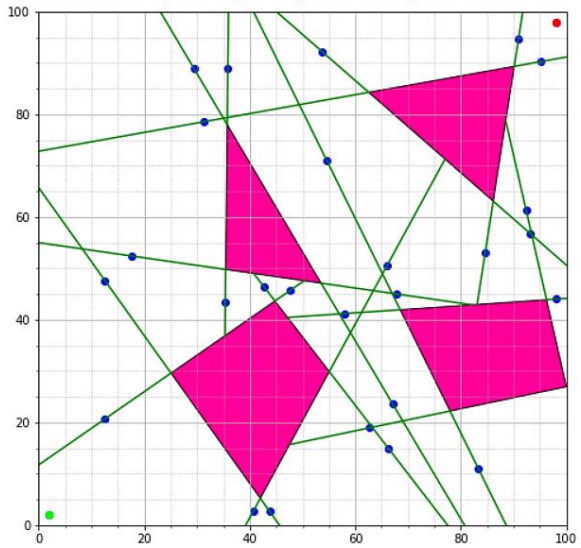

Рисунок 5 – Построение середин проведенных отрезков

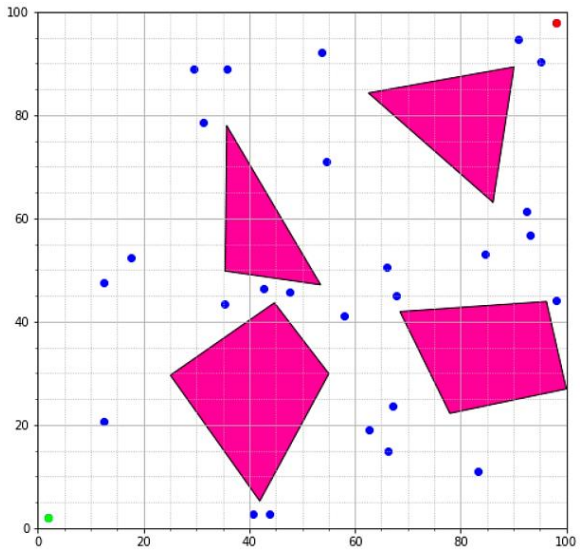

Рисунок 6 – Отображение вершин канального графа

Полученные вершины соединяются в связный граф, по следующему алгоритму.

1. Для удобства массив, содержащий координаты всех точек, сортируется по возрастанию координаты x.
2. Для каждой точки создается массив, содержащий координаты остальных точек, отсортированный по увеличению расстояния до них от рассматриваемой точки.
3. Каждая точка соединяется ребром с ближайшей к ней доступной точкой. Точка является доступной, если она не связана ребром другой точкой и если отрезок, соединяющий рассматриваемые точки не пересекает ни одно препятствие.
4. Следующей для рассмотрения выбирается точка, к которой было проведено ребро на предыдущем этапе.
5. Если среди всех точек нет доступных, то рассматриваемая точка соединяется ребром с уже построенным к данному моменту графом: проводится ребро, не пересекающее препятствия, к ближайшей точке, которая уже была соединена с какой-либо другой точкой. В

данном случае следующей для рассмотрения выбирается первая свободная точка в отсортированном по координатам массиве.

6. После соединения всех точек в граф, от первой точки проводится еще одно ребро, чтобы граф получился связным.

7. К полученному графу ребрами присоединяются точки, описывающие начальное и конечное положения робота. Данные точки соединяются с ближайшими к ним вершинами в графе.

Результат построения связного графа для рассматриваемого конфигурационного пространства представлен на Рисунке 7.

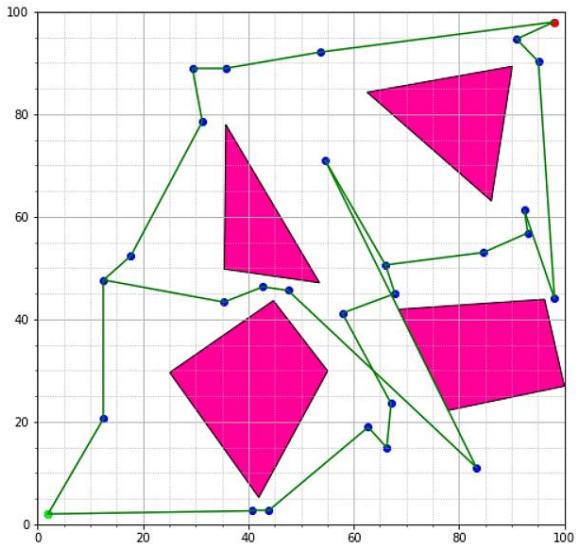

Рисунок 7 – Построение канального графа

Для поиска маршрута от стартовой до конечной точки был использован алгоритм Ли. Алгоритм Ли (алгоритм волновой трассировки) [4] – алгоритм поиска кратчайшего пути на планарном графе. Принадлежит к алгоритмам, основанным на методах поиска в ширину. Работа алгоритма включает в себя три этапа: инициализацию, распространение волны и восстановление пути. От стартовой вершины порождается шаг в соседнюю вершину, при этом проверяется не принадлежит ли она к ранее помеченным в пути вершинам.

При выполнении условия непринадлежности ее к ранее помеченным в пути вершинам, в атрибут вершины записывается число, равное количеству шагов от стартовой вершины, от стартовой вершины на первом шаге это будет 1. Каждая вершина, меченная числом шагов от стартовой вершины, становится стартовой и из нее порождаются очередные шаги в соседние вершины. Очевидно, что при таком переборе будет найден путь от начальной вершины к конечной, либо очередной шаг из любой порождённой в пути вершины будет невозможен. Восстановление кратчайшего пути происходит в обратном направлении: при выборе вершины от финишной вершины к стартовой на каждом шаге выбирается вершина, имеющая атрибут расстояния от стартовой на единицу меньше текущей вершины.

Результат работы данного этапа программы представлен на Рисунке 8. Итоговый маршрут робота из начальной точки в конечную изображен на Рисунке 9.

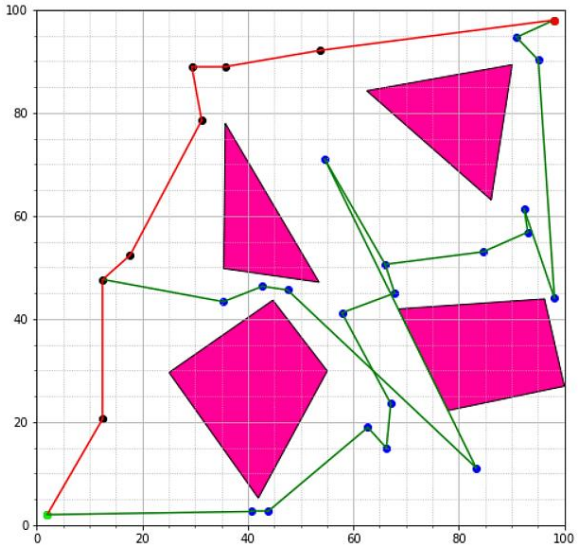

Рисунок 8 – Поиск пути в графе

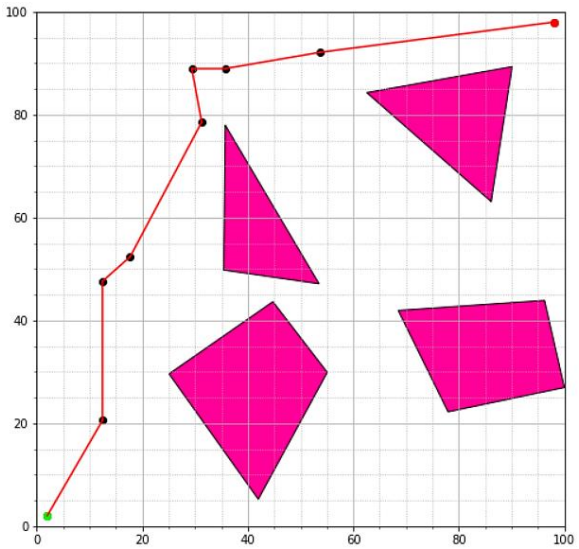

Рисунок 9 – Результат работы программы

Результаты

Разработанная программа позволяет находить маршрут движения робота из начальной точки в конечную, избегая столкновения с препятствиями. В Таблице 1 представлены результаты работы программы для конфигурационных пространств разной сложности (сложность определяется количеством вершин препятствий n).

На Рисунке 10 изображен график зависимости времени выполнения программы от количества вершин препятствий. По графику видно, что трудоемкость быстро возрастает с увеличением сложности конфигурационного пространства.

Таблица 1 – Результаты работы программы

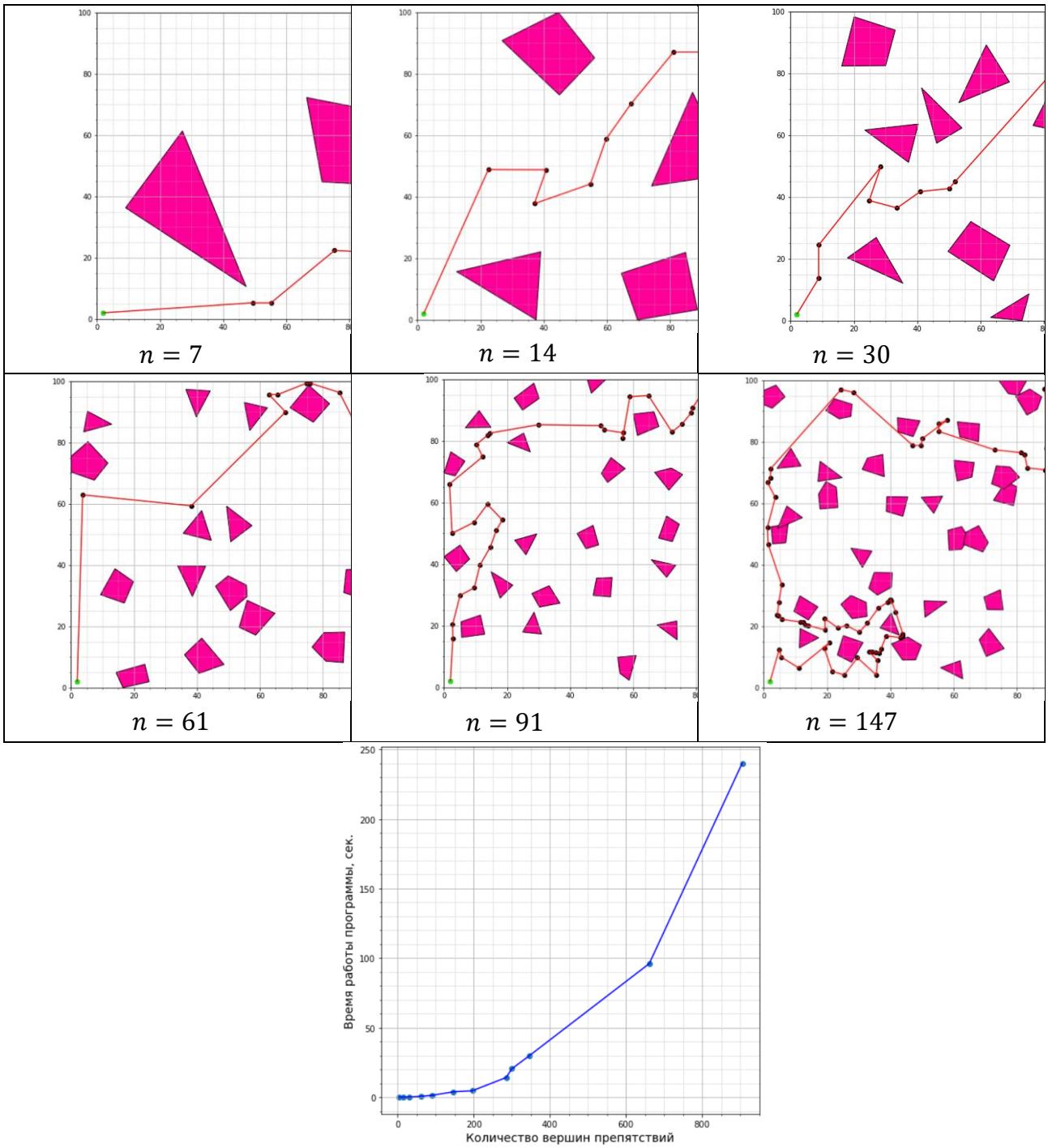

Рисунок 10 – Зависимость времени выполнения программы от сложности конфигурационного пространства

Список литературы

1. А.И. Губин. Построение маршрута с учетом динамических ограничений: Магистерская дис. – СПб., 2020. – 32 с. – Режим доступа: https://dspace.spbu.ru/bitstream/11701/26343/1/VKR.pdf.
2. К.А. Казаков, В.А. Семенов. Обзор современных методов планирования движения. Труды ИСП РАН, том 28, вып. 4, 2016, С. 241-294.

3. Steven M. LaValle. Planning algorithms. Copyright 2006. Cambridge University Press, 842 pages.
4. Алгоритм Ли. [Электронный ресурс], URL: https://ru.wikipedia.org/wiki/Алгоритм_Ли (дата обращения: 04.10.2022).

References

1. A.I. Gubin. Building a route taking into account dynamic constraints: Master's dis. – SPb., 2020. – 32 p. – Access mode: https://dspace.spbu.ru/bitstream/11701/26343/1/VKR.pdf.
2. K.A. Kazakov, V.A. Semenov. Review of modern methods of traffic planning. Trudy ISP RAS, vol. 28, vol. 4, 2016, S. 241-294.
3. Steven M. LaValle. Planning algorithms. Copyright 2006. Cambridge University Press, 842 pages.
4. Lee's algorithm. [Elektronnyi resurs], URL: https://ru.wikipedia.org/wiki/ Algoritm_Lee (date of access: 04.10.2022).