|  |  |
| --- | --- |
| Gerb-BMSTU_01 | **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 ЛАБОРАТОРНОЙ РАБОТЫ**

по дисциплине: «Автоматизация технологического проектирования»

|  |  |  |
| --- | --- | --- |
| Студент |  | Косенков Александр Александрович |
| Группа |  | РК6-32М |
| Тип задания |  | Лабораторная работа №1 |
| Тема лабораторной работы |  | Декомпозиция Морса при помощи окружностей |

Студент **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_Косенков А.А.\_**

*подпись, дата фамилия, и.о.*

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

*подпись, дата фамилия, и.о.*

Оценка \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

*Москва, 2022 г.*

**Оглавление**

[Задание на лабораторную работу 3](#_Toc124419687)

[Цель выполнения лабораторной работы 5](#_Toc124419688)

[Выполненные задачи 5](#_Toc124419689)

[Разработка исходных кодов 6](#_Toc124419690)

[Результаты численного решения ОДУ 9](#_Toc124419691)

[Автоматизация тестирования с использованием Gitlab CI 13](#_Toc124419692)

[Ответы на вопросы базовой части 15](#_Toc124419693)

[Заключение 15](#_Toc124419694)

[Список использованных источников 16](#_Toc124419695)

# Задание на лабораторную работу

Разработать программу разбиения конфигурационного пространства методом декомпозиции Морса при помощи окружностей. Препятствия создаются с помощью программы-генератора. Требуется определить зависимость времени выполнения программы от количества препятствий.

# Цель выполнения лабораторной работы

**Цель выполнения лабораторной работы** – ознакомление и изучение методов точной декомпозиции для решения задач планирования перемещений и реализация алгоритма декомпозиции Морса при помощи окружностей.

# Описание алгоритма

Идея декомпозиции Морса заключается в распространении от начальной точки определенного фронта (прямая линяя, окружность, спираль и т.д.) и анализ изменения его связности, которая заключается в количестве пересечений с препятствиями.

Формальное описание алгоритма декомпозиции Морса предполагает индуктивный сценарий (в соответствии с задачей за фронт принята окружность):

1. Задается шаг, с которым увеличивается радиус окружности от центра
2. Для разных радиусов от минимального до максимального с указанным шагом происходит подсчет пересечений с препятствиями
3. Запоминаются радиусы окружностей, при которых количество пересечений (связность) меняется
4. Формируется граф свободных областей путем анализа окружностей по увеличению их радиуса, начиная от левой границы

Однако, поиск пересечения со всеми препятствиями может быть крайне ресурсозатратен при росте их количества, а выбор шага является нетривиальной задачей, поскольку при слишком маленьком шаге сложность алгоритма значительно возрастает, а при маленьком падает точность и релевантность свободных областей, так как в них могут попадать части препятствий.

По этой причине более оптимальным может быть дедуктивный сценарий, который предполагает выполнение описанных в индуктивном алгоритме действий на основе априорных знаний о задаче. Так, места нарушения связности фронта являются критическими точками, определение которых является основной целью дедуктивного алгоритма (вместо поиска по шагам).

С практической точки зрения критические точки являются развилками для покрывающего робота (рис. 1). Когда робот покрывает определенный сектор и встречает критическую точку, он вынужден пойти только в один из следующих секторов. Для покрытия обоих ему придется вернутся в начальный.

Так как в рамках решаемой задачи конфигурационное пространство состоит из многоугольников (кривые отсутствуют) — критическими точками в большинстве своем являются их вершины (рис. 1, 2, 4) или точка на грани в частном случае (рис. 3). Далее рассмотрены различные случаи, которые возникают при анализе критических точек.

1. Вершина препятствия с наименьшим радиусом влечет за собой разделение предыдущего сектора на два новых:

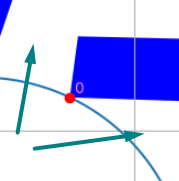


Рис. . Пример критической точки.

1. Вершина препятствия с наибольшим радиусом влечет за собой слияние двух предыдущих секторов в один:

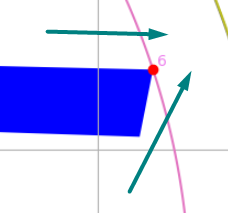


Рис. . Пример критической точки.

1. Минимальный радиус соприкосновения с препятствием не в вершине, а на грани. Данный случай можно обнаружить путем проверки количества пересечений окружности с полигоном от вершины с наименьшим радиусом (если пересечений больше одного, то внутрь окружности попала область препятствия, а значит минимальный радиус в точке касания):

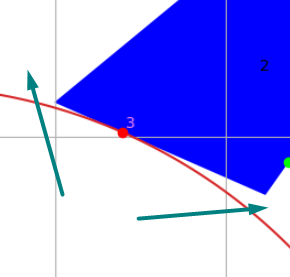


Рис. . Пример критической точки.

1. Выпуклые внутрь области препятствий также требуют отдельного анализа, поскольку могут также повлечь развилку для покрывающего робота:

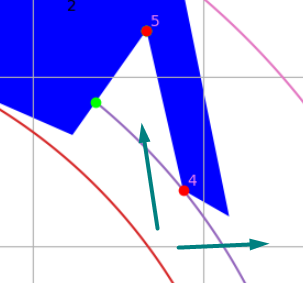


Рис. . Пример критических точек на выпуклой внутрь области препятствия.

1. Критической точкой не является точка на границе поля, если они минимальная или максимальная, поскольку в таком случае не нарушается связность фронта и не возникает развилок:

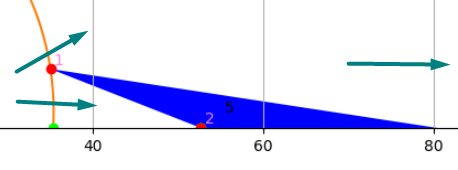


Рис. . Пример частного случая анализа критических точек.

Исключение: препятствие касается границы поля только в одной точке. В таком случае какую-либо из зон обязательно придется ограничить именно этой точкой.

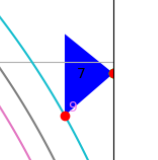


Рис. . Пример частного случая анализа критических точек.

1. Критической точкой не является та, границы дуги которой совпадают с вершиной полигона. Это обусловлено тем, что робот без труда сможет покрыть траекторией окружности область без развилок:

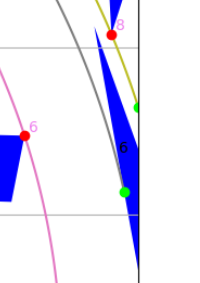
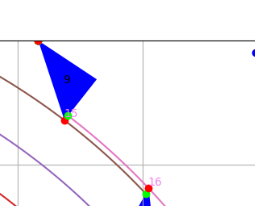
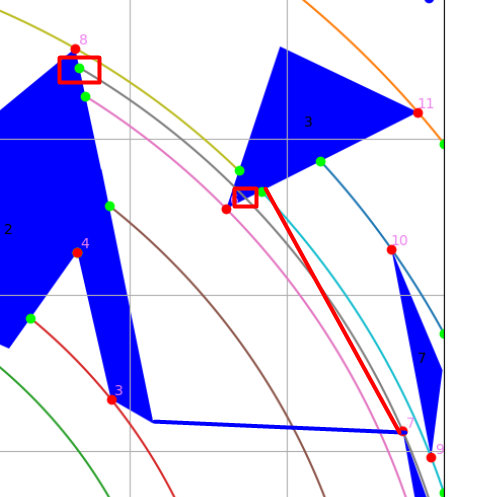


Рис. 7. Пример частного случая анализа критических точек.

После получения всех критических точек возможно

Точка позже финальной дуги





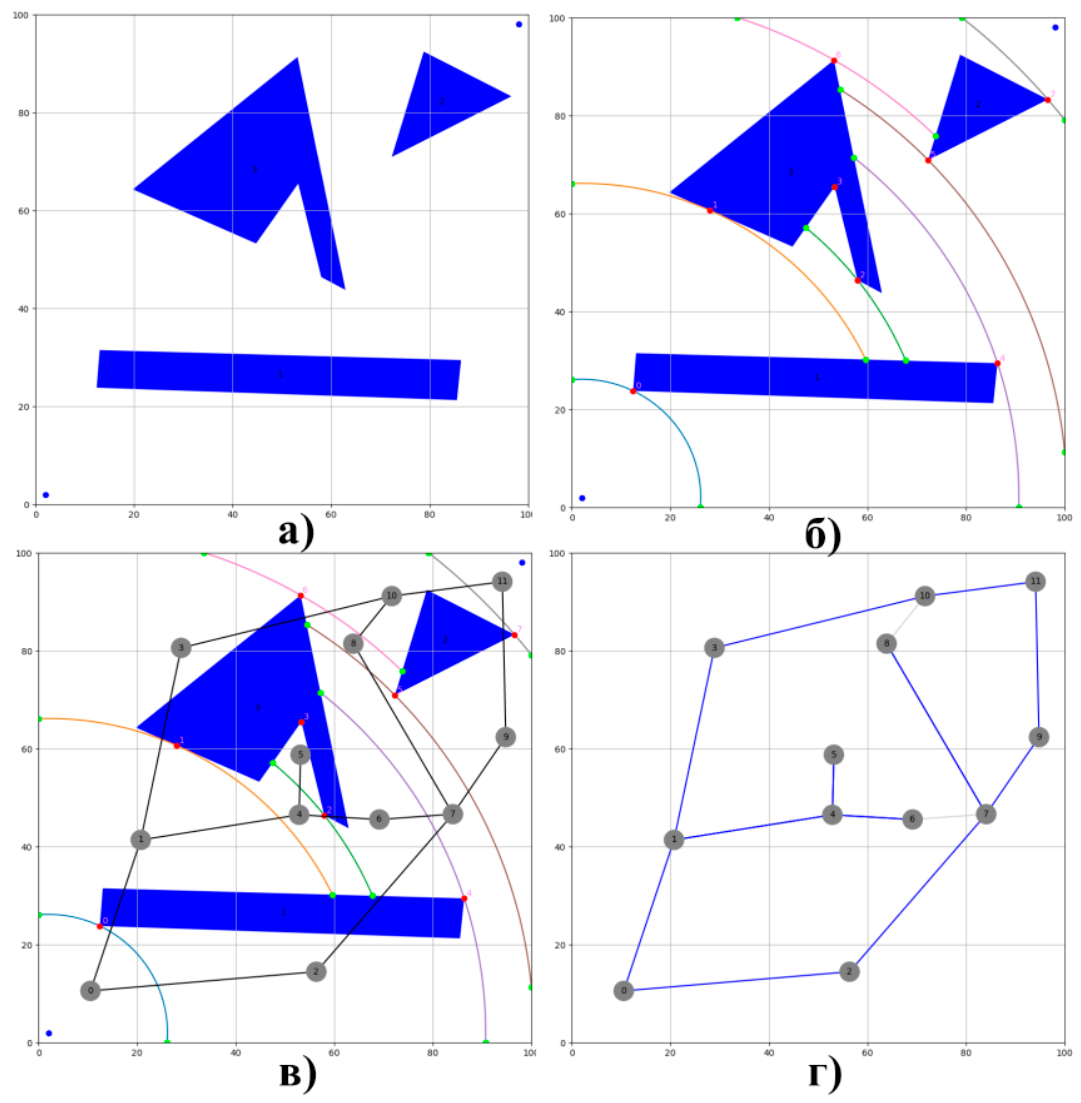


Рис. 1. Демонстрация работы программы для тестового примера. (а) — исходное конфигурационное пространство, (б) — декомпозиция пространства дугами, (в) — граф свободных секторов, (г) — граф с отображением ребер обхода.

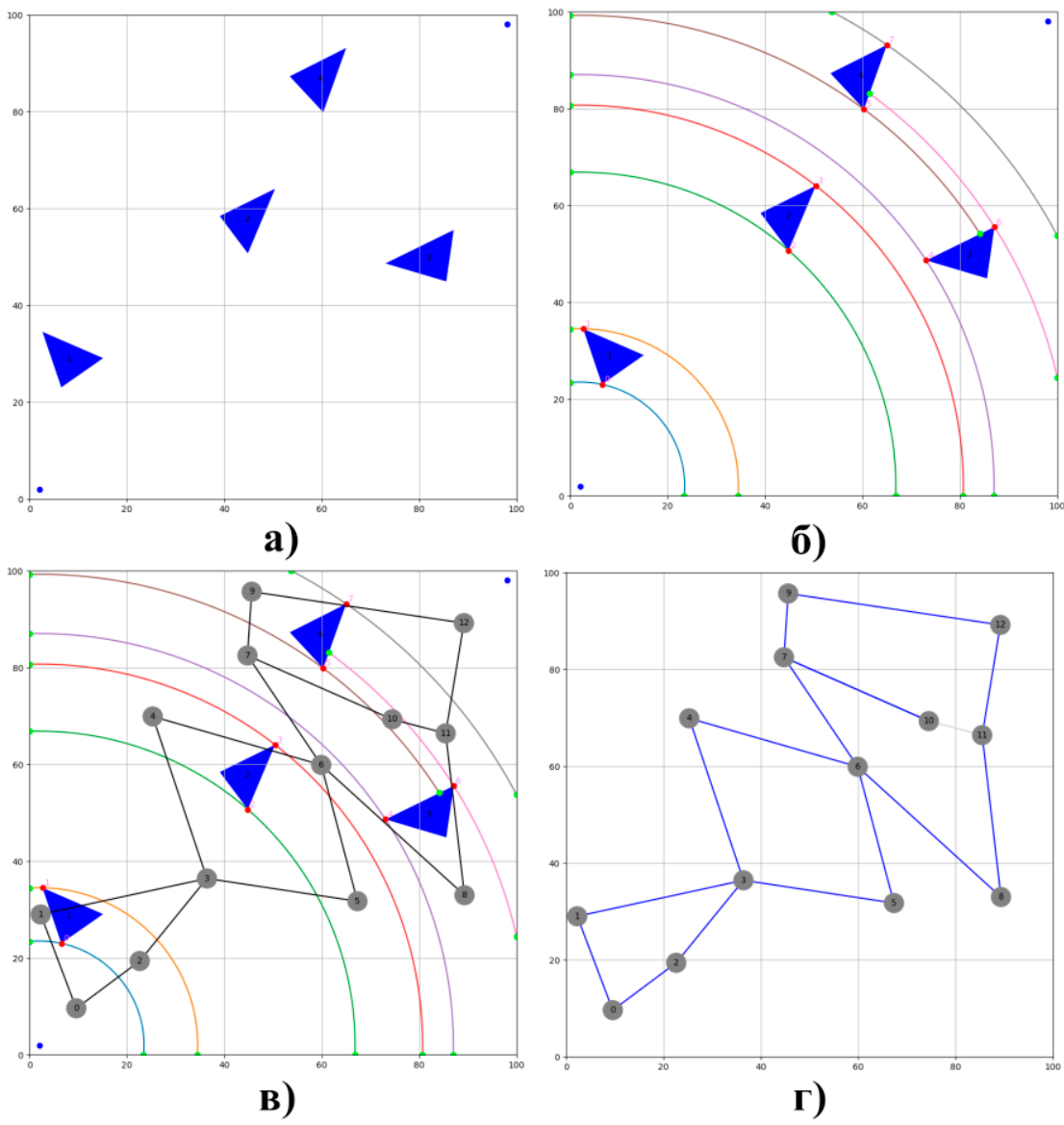


Рис. . Демонстрация работы программы для 4-х препятствий (размер 9). (а) — исходное конфигурационное пространство, (б) — декомпозиция пространства дугами, (в) — граф свободных секторов, (г) — граф с отображением ребер обхода.

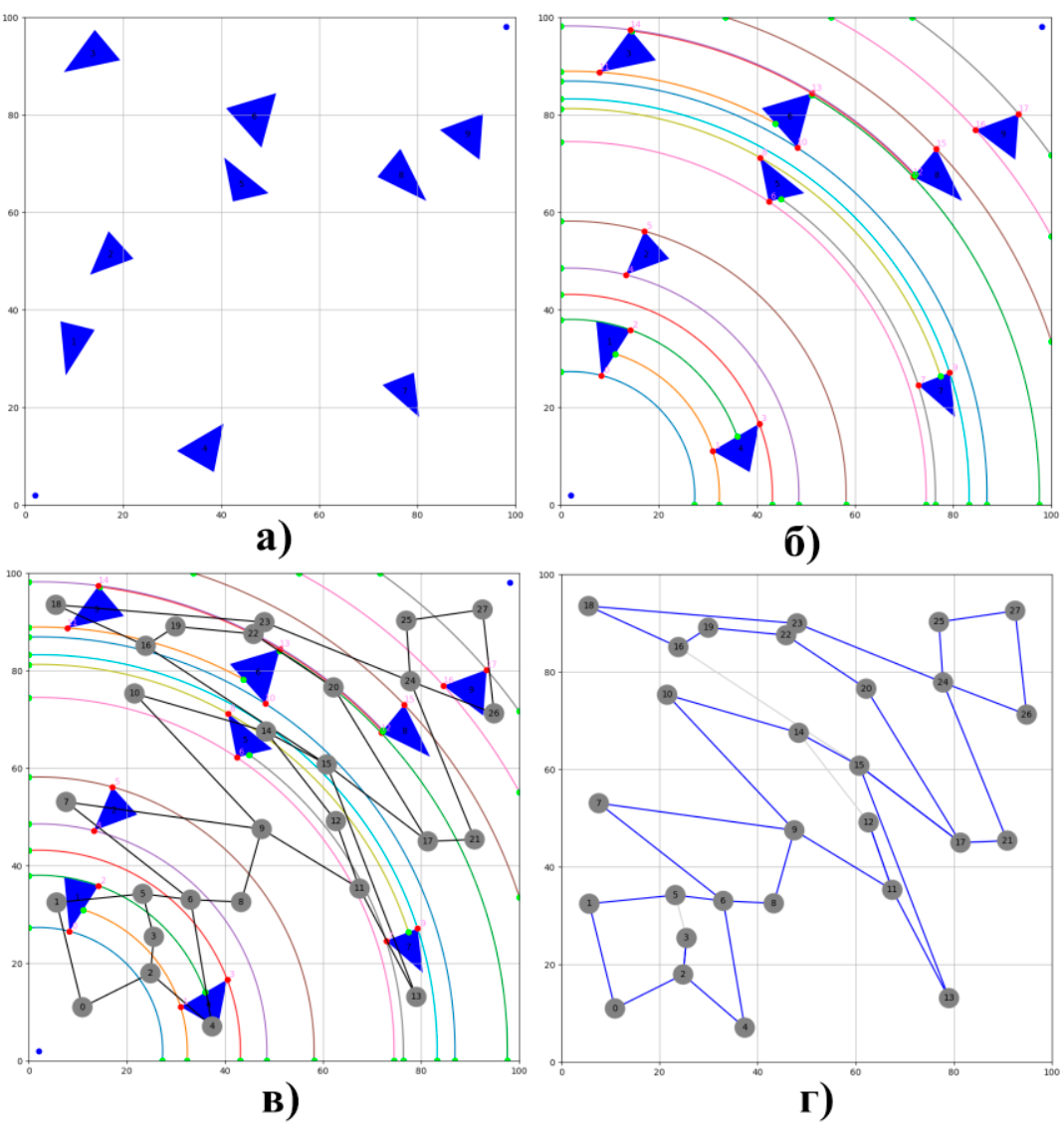


Рис. . Демонстрация работы программы для 9-и препятствий (размер 7). (а) — исходное конфигурационное пространство, (б) — декомпозиция пространства дугами, (в) — граф свободных секторов, (г) — граф с отображением ребер обхода.

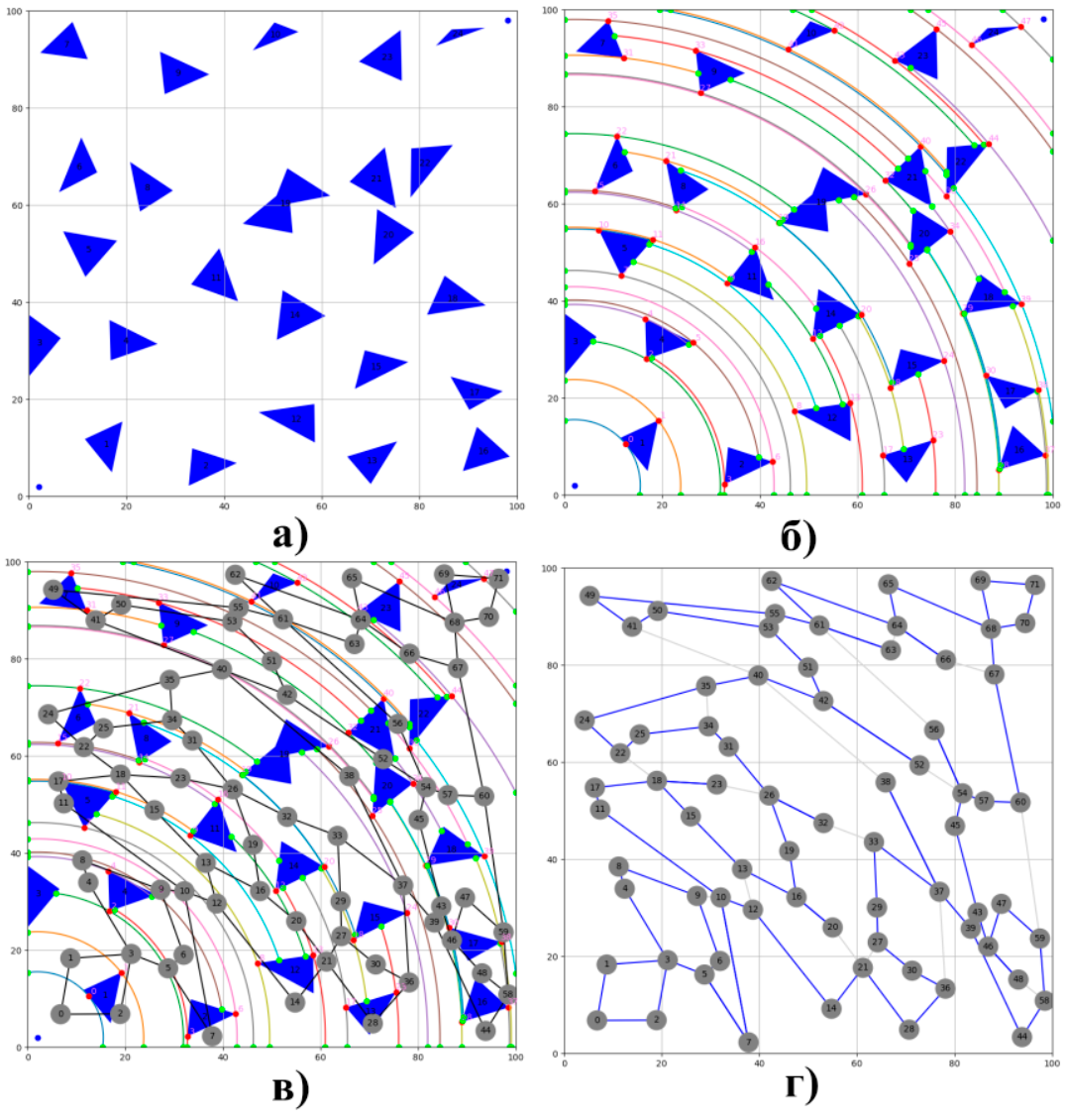


Рис. . Демонстрация работы программы для 25-и препятствий (размер 7). (а) — исходное конфигурационное пространство, (б) — декомпозиция пространства дугами, (в) — граф свободных секторов, (г) — граф с отображением ребер обхода.

Изображение выглядит как стрела

Автоматически созданное описание

Рис. . Демонстрация работы программы для 64-х препятствий (размер 6). (а) — исходное конфигурационное пространство, (б) — декомпозиция пространства дугами, (в) — граф свободных секторов, (г) — граф с отображением ребер обхода.

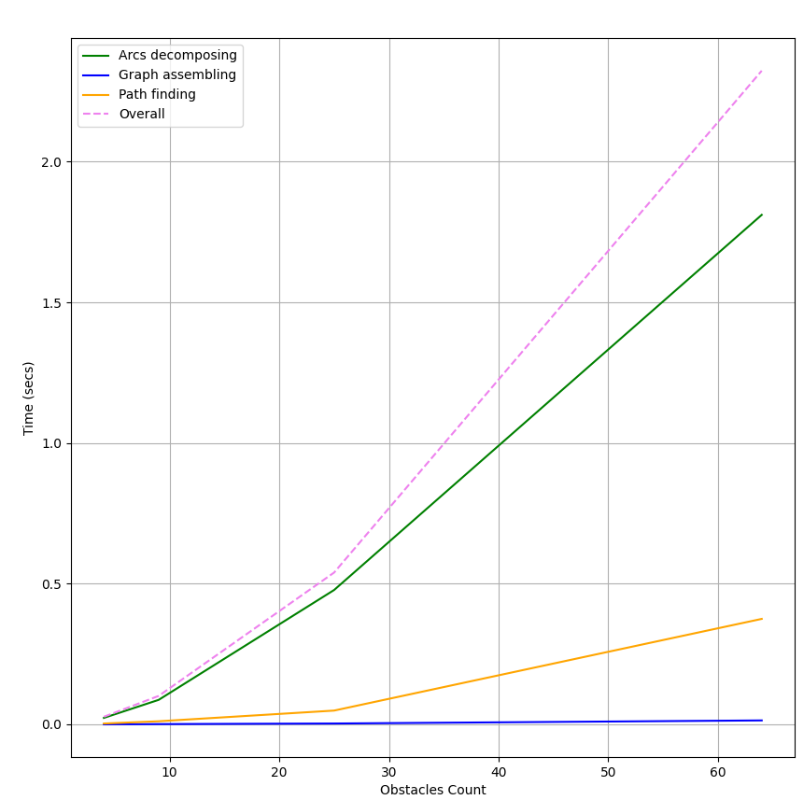


Рис. . Сравнение временной эффективности в зависимости от количества препятствий в конфигурационном пространстве.

# Заключение

В рамках лабораторной работы 3 на примере численного решения заданного ОДУ (1) методом Адамса-Башфорта-Моултона был изучен и реализован процесс автоматизации тестирования программного кода с использованием системы непрерывной интеграции GitLab CI.

# Список использованных источников

1. **Косенков, А.А.** Комплект документации к лабораторной работе 3. URL: [https://sa2systems.ru:88/rk6\_student/edupmi/tree/2022\_rk6\_32m\_kosenkovaa/lab3](https://sa2systems.ru:88/rk6_student/edupmi/tree/2022_rk6_32m_kosenkovaa/lab31)
2. **Першин А.Ю.** *Лекции по вычислительной математике (черновик).* [Электронный ресурс] // Кафедра РК6 (Системы автоматизированного проектирования). МГТУ им. Н.Э. Баумана, Москва, 2020 г., 142 с.