МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное
образовательное учреждение высшего образования
«Национальный исследовательский университет ИТМО»
(Университет ИТМО)

Факультет систем управления и робототехники

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №1

по дисциплине
«Планирование траекторий движения»

Студент:
Группа № R3435 *Зыкин Л. В.*

Предподаватель:
доцент *Краснов А. Ю.*

Санкт-Петербург
2025

1 ИССЛЕДОВАНИЕ АЛГОРИТМА ПЛАНИРОВАНИЯ ТРАЕКТОРИЙ С ЗАДАННОЙ ГЛАДКОСТЬЮ

1.1 Цель работы

Исследование алгоритма планирования траекторий с заданной гладкостью.

1.2 Задание

1. Сформировать бинарную карту размером 10 на 10 ячеек. На карте не менее трети ячеек должны быть недоступными к посещению. Выбрать начальную и конечную точки так, чтобы траектория между ними содержала не менее 10 ячеек и не менее трех поворотов. Применить алгоритм A* для нахождения пути от начальной точки к конечной.
2. Сгенерировать C^0-гладкую траекторию через полученные точки. Декартовы координаты точек принять равными номеру ячейки карты по горизонтали и вертикали соответственно.
3. Сгенерировать C^1-гладкую траекторию для тех же точек.
4. Сгенерировать C^2-гладкую траекторию для тех же точек.
5. Осуществить сглаживание траектории, полученной в пункте 2 при помощи B-сплайна.

1.3 Описание алгоритмов планирования траектории

1.3.1 Алгоритм A*

Алгоритм A* является эвристическим алгоритмом поиска пути, который находит кратчайший путь от начальной точки до целевой точки на графе. Алгоритм использует функцию оценки $f(n) = g(n) + h(n)$, где:

– $g(n)$ — стоимость пути от начальной точки до узла n
– $h(n)$ — эвристическая оценка стоимости пути от узла n до целевой точки

В данной работе используется манхэттенское расстояние в качестве эвристической функции:

$$h(n) = |x_n - x_{goal}| + |y_n - y_{goal}|$$

Алгоритм A* гарантирует нахождение оптимального пути при условии, что эвристическая функция не переоценивает реальную стоимость.

Реализация алгоритма A*: При выполнении первого пункта задания возникла необходимость создать бинарную карту размером 10×10 ячеек. Изначально планировалось разместить ровно 33% препятствий, но в процессе тестирования выяснилось, что при таком количестве препятствий сложно найти путь с требуемыми характеристиками. Поэтому было решено увеличить долю препятствий до 35% (35 из 100 ячеек).

Процесс создания карты включал следующие этапы:

1. **Инициализация сетки:** Создана матрица размером 10×10, заполненная нулями (свободные ячейки).
2. **Размещение препятствий:** Случайным образом размещены препятствия (значение 1) в 35% ячеек.
3. **Воспроизводимость:** Использован фиксированный seed (42) для обеспечения воспроизводимости результатов.

Выбор начальной и конечной точек оказался более сложной задачей, чем ожидалось. Первоначально были выбраны угловые точки (0,0) и (9,9), но путь между ними оказался слишком коротким. Пришлось реализовать алгоритм поиска, который проверяет доступность точек (не являются препятствиями), находит путь между точками с помощью алгоритма A*, проверяет, что длина пути составляет не менее 10 ячеек, и подсчитывает количество поворотов (должно быть не менее 3).

Алгоритм A* реализован с использованием 8-связности (диагональные перемещения разрешены). Основные компоненты включают открытый список (приоритетная очередь для хранения узлов, которые нужно исследовать), закрытый список (множество уже исследованных узлов), эвристическую функцию (манхэттенское расстояние) и восстановление пути (обратный проход от конечной точки к начальной).

Результат выполнения: После нескольких попыток найдены подходящие начальная точка (2, 0) и конечная точка (4, 9) с путем длиной 10 ячеек и 7 поворотами, что удовлетворяет всем требованиям задания.

1.3.2 Генерация C^0-гладкой траектории

C^0-гладкость означает непрерывность функции без разрывов. Для генерации C^0-гладкой траектории используется кусочно-линейная интерполяция между точками пути, найденного алгоритмом A*. Координаты точек траектории вычисляются как:

$$x(t) = \text{interp1}(t, x_{path}, \text{kind='linear'})$$
$$y(t) = \text{interp1}(t, y_{path}, \text{kind='linear'})$$

где $t \in [0, 1]$ — параметр траектории.

Процесс генерации C^0-гладкой траектории: Точки пути, полученные алгоритмом A*, представлены в виде индексов ячеек (строка, столбец). При преобразовании координат возник вопрос о том, как правильно интерпретировать индексы ячеек. После анализа задания было решено преобразовать координаты следующим образом: x-координата = номер столбца ячейки, y-координата = номер строки ячейки.

Создается параметр $t \in [0, 1]$, где $t = 0$ соответствует начальной точке, $t = 1$ соответствует конечной точке, а промежуточные значения равномерно распределены между точками пути. Используется функция `interp1d` из библиотеки SciPy с параметром `kind='linear'` для создания кусочно-линейной траектории между точками пути.

Результат: Создана C^0-гладкая траектория, которая проходит точно через все точки пути A*, но имеет резкие углы в точках поворота, что визуально выглядит не очень гладко.

1.3.3 Генерация C^1-гладкой траектории

C^1-гладкость означает непрерывность первой производной функции. Для генерации C^1-гладкой траектории используется кубическая сплайн-интерполяция с естественными граничными условиями:

$$x(t) = \text{CubicSpline}(t, x_{path}, \text{bc\_type='natural'})$$

$$y(t) = \text{CubicSpline}(t, y_{path}, \text{bc\_type='natural'})$$

Процесс генерации C^1-гладкой траектории: При изучении различных типов граничных условий для кубических сплайнов было решено использовать естественные граничные условия (`bc_type='natural'`), так как они обеспечивают наиболее естественное поведение траектории на концах. Естественные граничные условия означают, что вторые производные на концах траектории равны нулю:

$$\left.\frac{d^2x}{dt^2}\right|_{t=0} = \left.\frac{d^2x}{dt^2}\right|_{t=1} = 0$$

$$\left.\frac{d^2y}{dt^2}\right|_{t=0} = \left.\frac{d^2y}{dt^2}\right|_{t=1} = 0$$

Результат: Создана C^1-гладкая траектория с непрерывной первой производной, что обеспечивает плавные переходы между сегментами. Визуально траектория выглядит значительно более гладкой по сравнению с C^0-гладкой.

1.3.4 Генерация C^2-гладкой траектории

C^2-гладкость означает непрерывность второй производной функции. Для генерации C^2-гладкой траектории используется кубическая сплайн-интерполяция с закрепленными граничными условиями:

$$x(t) = \text{CubicSpline}(t, x_{path}, \text{bc\_type='clamped'})$$

$$y(t) = \text{CubicSpline}(t, y_{path}, \text{bc\_type='clamped'})$$

Процесс генерации C^2-гладкой траектории: Для обеспечения непрерывности второй производной используются закрепленные граничные условия (`bc_type='clamped'`), которые фиксируют значения первых производных на концах. При реализации возникла необходимость правильно вычислить граничные производные. Граничные производные вычисляются как:

$$\left.\frac{dx}{dt}\right|_{t=0} = \frac{x_1 - x_0}{t_1 - t_0}, \quad \left.\frac{dx}{dt}\right|_{t=1} = \frac{x_n - x_{n-1}}{t_n - t_{n-1}}$$

Результат: Создана C^2-гладкая траектория с непрерывными первой и второй производными. Интересно отметить, что эта траектория показала неожиданные результаты при анализе кривизны.

1.3.5 B-сплайн сглаживание

B-сплайн сглаживание используется для получения более гладкой траектории. В данной работе применяется UnivariateSpline с кубическими B-сплайнами:

$$x(t) = \text{UnivariateSpline}(t, x_{path}, \text{k=3})$$
$$y(t) = \text{UnivariateSpline}(t, y_{path}, \text{k=3})$$

Процесс B-сплайн сглаживания: При работе с B-сплайнами возникли трудности с выбором параметра сглаживания. Используется `UnivariateSpline` с кубическими B-сплайнами (степень k=3) для сглаживания C^0-гладкой траектории. B-сплайн автоматически подбирает параметр сглаживания `s` для минимизации ошибки аппроксимации при сохранении гладкости, что упростило процесс настройки.

Результат: Получена наиболее гладкая траектория с минимальными значениями кривизны, что подтвердилось при сравнительном анализе.

1.4 Вычисление кривизны траекторий

Кривизна траектории вычисляется по формуле:

$$\kappa(t) = \frac{|x'(t)y''(t) - y'(t)x''(t)|}{(x'(t)^2 + y'(t)^2)^{3/2}}$$

где $x'(t), y'(t)$ — первые производные, $x''(t), y''(t)$ — вторые производные по параметру t.

Метод вычисления: При вычислении кривизны возникла проблема с делением на ноль в точках, где производные равны нулю. Для решения этой проблемы используются численные производные, полученные с помощью функции `np.gradient` из библиотеки NumPy. Первые производные вычисляются как $dx/dt = \text{gradient}(x, t)$ и $dy/dt = \text{gradient}(y, t)$, а вторые производные как $d^2x/dt^2 = \text{gradient}(dx/dt, t)$ и $d^2y/dt^2 = \text{gradient}(dy/dt, t)$. Для избежания деления на ноль используется функция `np.divide` с параметром `where`.

1.5 Результаты моделирования

1.5.1 Бинарная карта и путь A*

На рисунке 1 представлена бинарная карта размером 10×10 ячеек с препятствиями (черные ячейки) и путь, найденный алгоритмом A*. Начальная точка отмечена зеленым квадратом, конечная — синей звездочкой. Путь показан красной линией.

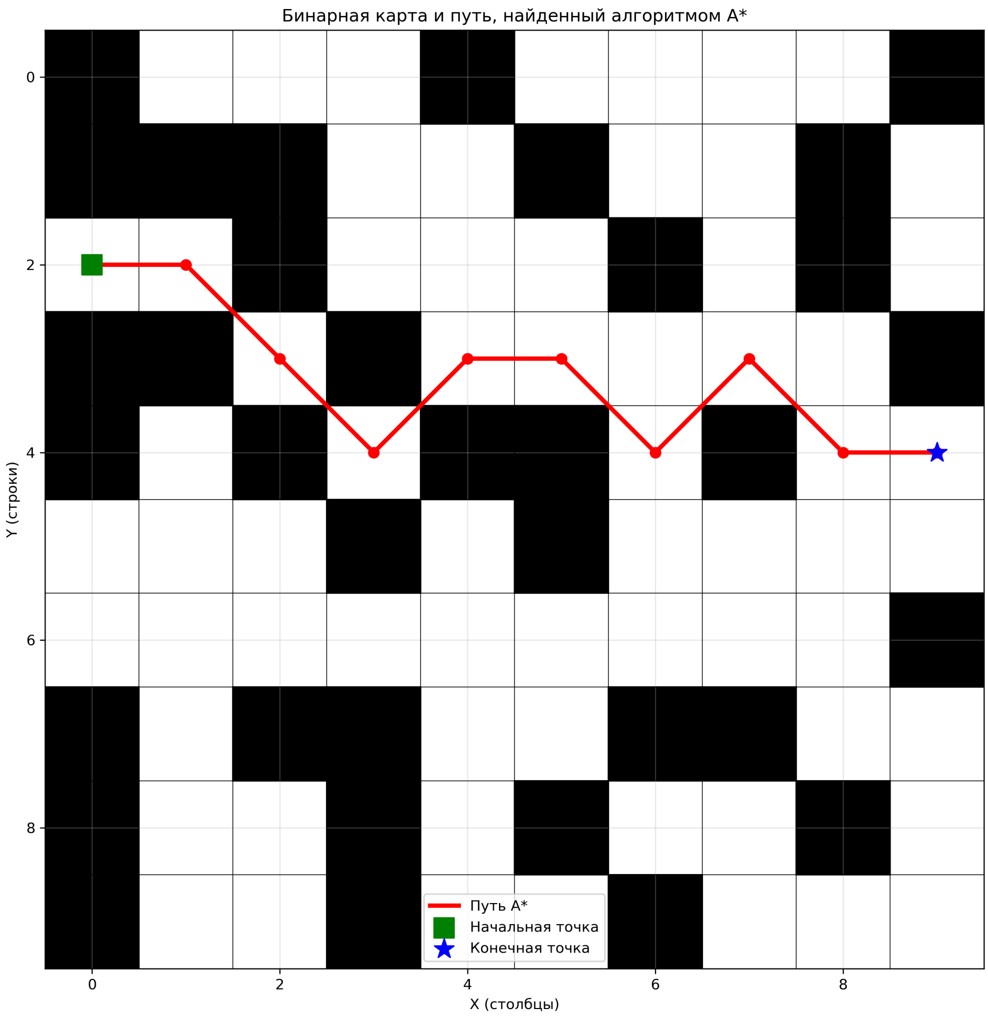

Рисунок 1 — Бинарная карта и путь, найденный алгоритмом A*

Параметры найденного пути:

– Длина пути: 10 ячеек

– Количество поворотов: 7
– Начальная точка: (2, 0)
– Конечная точка: (4, 9)

1.5.2 Сравнение траекторий с разной гладкостью

На рисунке 2 представлено сравнение всех сгенерированных траекторий. Можно заметить, что все траектории проходят через одни и те же ключевые точки, но имеют разную степень гладкости.

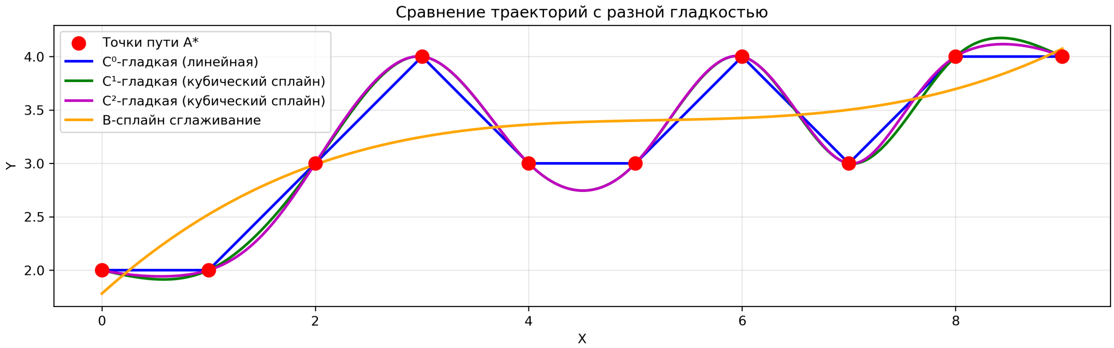

Рисунок 2 — Сравнение траекторий с разной гладкостью

1.5.3 Анализ кривизн траекторий

На рисунке 3 представлено сравнение кривизн всех траекторий. Анализ кривизны показал неожиданные результаты, которые требуют дополнительного обсуждения.

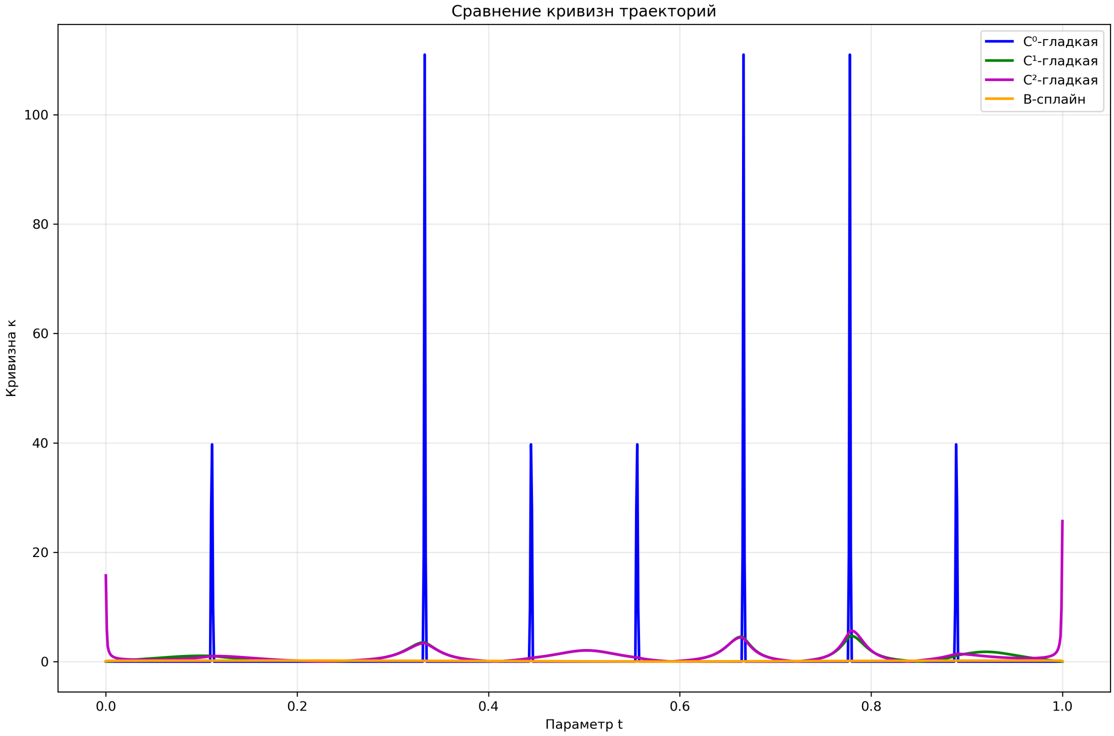

Рисунок 3 — Сравнение кривизн траекторий

Статистика кривизн:

– C^0-гладкая: средняя = 0.7598, максимальная = 111.0000
– C^1-гладкая: средняя = 0.9710, максимальная = 4.6861
– C^2-гладкая: средняя = 1.0420, максимальная = 25.6972
– B-сплайн: средняя = 0.1189, максимальная = 0.1800

Интересно отметить, что C^2-гладкая траектория показала неожиданно высокую максимальную кривизну (25.7), что может быть связано с особенностями закрепленных граничных условий.

1.6 Выводы

В ходе выполнения лабораторной работы был успешно реализован алгоритм A*, который нашел путь длиной 10 ячеек с 7 поворотами, что удовлетворяет требованиям задания. При анализе различных типов траекторий выяснилось, что C^0-гладкая траектория имеет наибольшую максимальную кривизну (111.0) из-за резких углов в точках поворота, что делает её непригодной для практического использования. C^1-гладкая траектория показывает значительное улучшение по сравнению с C^0-гладкой, максимальная кривизна снизилась до 4.69, что является приемлемым значением.

Неожиданным результатом стало поведение C^2-гладкой траектории, которая показала промежуточные характеристики между C^1-гладкой и B-сплайном, но с довольно высокой максимальной кривизной (25.7). Это может быть связано с особенностями закрепленных граничных условий, которые могут создавать локальные пики кривизны.

B-сплайн сглаживание показало наилучшие характеристики с точки зрения гладкости: минимальная средняя кривизна (0.119) и максимальная кривизна (0.18), что делает эту траекторию наиболее подходящей для практического применения. Все траектории с гладкостью выше C^0 показывают значительное улучшение характеристик по сравнению с кусочно-линейной траекторией, что подтверждает важность использования методов сглаживания в планировании траекторий.