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 *Робототехника и комплексная автоматизация*

КАФЕДРА *Системы автоматизированного проектирования (РК-6)*

РАСЧЕТНО-ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

К ВЫПУСКНОЙ КВАЛИФИКАЦИОННОЙ РАБОТЕ

НА ТЕМУ

«*Планирование перемещений в среде с препятствиями при помощи метода Искусственных потенциальных полей*»

Студент РК6-82Б _____ М.Н. Мудриченко
(Группа) (подпись, дата) (инициалы и фамилия)

Руководитель ВКР _____ А.Н. Божко
(подпись, дата) (инициалы и фамилия)

Нормоконтролер _____ С.В. Грошев
(подпись, дата) (инициалы и фамилия)

Москва, 2023 г.

РЕФЕРАТ

Работа посвящена изучению задачи планирования перемещения робота в пространстве с препятствиями. Рассматриваются методология использования виртуальных полей для определения желаемых траекторий движения робота.

Тип работы: выпускная квалификационная работа.

Тема работы (проект темы): решение задачи планирования перемещений методом искусственных потенциальных полей.

Объект исследований: задача планирования перемещений и метод искусственных потенциальных полей.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ 4
1 Постановка задачи 6
1.1 Постановка решаемой задачи 6
1.2 Конфигурационные пространства 8
2 Способы решения задачи планирования движения 13
2.1 Классификация и обзор методов планирования движения 13
2.1.1 Оптимизационные методы 13
2.1.2 Методы на основе графов 14
2.1.3 Методы на основе клеточной декомпозиции 15
2.1.4 Методы на основе интеллектуальных технологий 16
2.1.5 Методы иерархии подцели 17
2.1.6 Вероятностные методы 18
2.2 Метод искусственных потенциальных полей 19
2.3 Описание работы алгоритм МИПП 21
2.4 Потенциалы отталкивания и притяжения 23
2.5 Метод поиска локального минимума BFS 27
3 Программная реализация 28
3.1 Реализация метода МИИП 28
3.2 Тестирование работы программы 31
ЗАКЛЮЧЕНИЕ 42
СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 43
Приложение А 46

ВВЕДЕНИЕ

Современный объём и сложность производства не позволяют обходиться без автоматизации технологически сложных процессов. Введение в предприятие роботизированных механизмов не только ускоряет выполнение задач, но также позволяет сократить трудозатраты [1].

Задача планирования перемещения является неотъемлемой частью автоматизации производственных и мобильных роботов. В последнее время, интерес к ней заметно возрос в связи развитием средств компьютерного моделирования, инновационных робототехнических технологий, формирования таких дисциплин как роботизированная хирургия, автоматизация сборки продуктов, организация транспортных потоков в мегаполисах [2].

Автоматизация также занимает немалую часть в сфере маркетинга и развлечений. Люди, занимающиеся графическим моделированием и виртуальной реальностью, пользуются технологией для анимаций создаваемых объектов.

С технологией планирования движения и соответствующей системой датчиков, роботы могут быстро адаптироваться к неожиданным изменениям в окружающей среде, обрабатывать и устранять ошибки, совершенные в моделировании.

Под задачей планирования пути принято понимать перемещение робота от начальной до конечной точки. При этом, спланированный путь должен обеспечивать движение с обходом препятствий, быть оптимальным, удовлетворять различным динамическим и кинематическим ограничениям [3].

Существует множество различных методов решения задачи планирования движения. Среди них можно выделить метод искусственных потенциальных полей (МИПП). Данный метод изначально разрабатывался для навигации мобильных роботов и обхода препятствий в режиме реального времени. МИПП основан на аналогии с движением заряженной частицей в электростатическом поле. Препятствия поля генерируют силу отталкивания, целевая точка маршрута

генерирует силу притяжения и представляет отрицательный заряд. За заряженную частицу принимается мобильный работ [4].

Отличительной особенностью алгоритма является возможность использования для пространств высокой размерности. Что позволяет решать задачи планирования пути в пространстве повышенной плотности [6].

Минусами данного алгоритма можно считать экспоненциальный рост зависимости времени вычислений от плотности при выборе точности решения.

Для составления решения МИПП пространство разбивается на сетку на двумерной декартовой плоскости. Каждой ячейке сетки ставится характерное значение, называемое потенциалом, показывающее уровень “уверенности” алгоритма в присутствии препятствия в данной ячейке. Путь находится одним из алгоритмов поиска локального минимума. Одним из самых простых способов решения данной проблемы является алгоритм Best first search (BFS) [5,7].

Данная выпускная квалификационная работа нацелена на решение задачи планирования перемещений методом искусственных потенциальных полей. В ходе выполнения работы были определены следующие задачи:

1) исследовать области применения, постановку и актуальность задачи планирования перемещений;
2) провести анализ различных методов и классов методов;
3) реализовать алгоритм МИПП на языке Python, модифицировать алгоритм, создав зависимость шага от плотности препятствий в пространстве;
4) провести тестирование алгоритма МИПП в различных условиях загруженности конфигурационного пространства, провести сравнительный анализ времени работы алгоритма в средах повышенной и пониженной плотности.

1 Постановка задачи

1.1 Постановка решаемой задачи

Задача планирования перемещения является вычислительной задачей, решением которой является свободный от столкновений допустимый путь от начальной до целевой конфигурации.

Задача задается описанием геометрии, задающей форму и положение моделируемого устройства (манипулятора, мобильного робота), его окружения (препятствий, создающих помехи при движении устройства), а также начальной и целевой конфигураций устройства [8].

Применение задачи включается в поиске оптимальных маршрутов робота, планировании траектории роботизированного манипулятора, поиске минимального расстояния между двумя точками на карте.

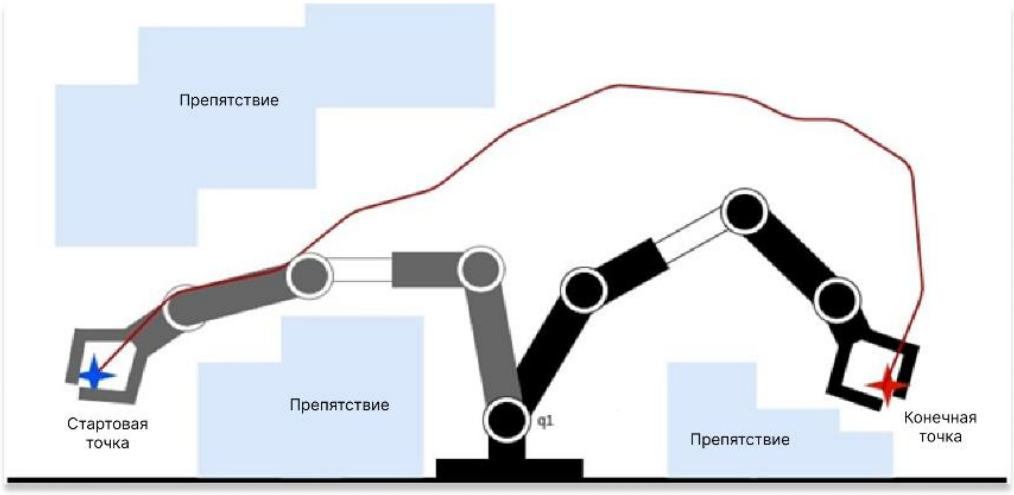

Рисунок 1 – Траектория автономно запрограммированного робота манипулятора.

Одной из основных проблем задачи является зависимость между вычислительной точностью и скоростью вычислений.

Поскольку планирование маршрута, как правило, допускает бесконечное множество решений (хотя может не существовать ни одного решения), иногда данную задачу формулируют в постановке оптимизационной задачи с целевой

функцией, соответствующей минимальной длине маршрута или максимальной удаленности перемещаемого объекта от препятствий. На практике, поиск пути даже в простых сценах с относительно небольшим количеством препятствий становится трудноразрешимой задачей, если перемещаемый объект имеет сложную геометрию или высокое число степеней свободы. В современных индустриальных приложениях часто требуется моделировать поведение сложных кинематических систем с шестью и более степенями свободы в статическом или динамическом окружении, насчитывающим тысячи препятствий. Кроме того, следует учитывать физические свойства перемещаемого объекта, учитывать габариты и просчитывать максимально допустимые риски столкновения [1].

С учётом перечисленных трудностей и невозможности вычисления траекторий с идеальной точностью, предпочтение в современных методах планирования движения отдают эвристическим методам решения.

В данной работе рассматривается метод искусственных потенциальных полей, находящий оптимальную траекторию в независимости от размера поля и плотности препятствий. Решение задачи рассматривается на двумерном пространстве. Мобильный робот, представляется в виде точки, препятствия в виде простых геометрических объектов. Из точки начала робот должен попасть в конечную [4].

На рисунке 2 показано схематичное решение задачи планирования движения в двумерном пространстве.

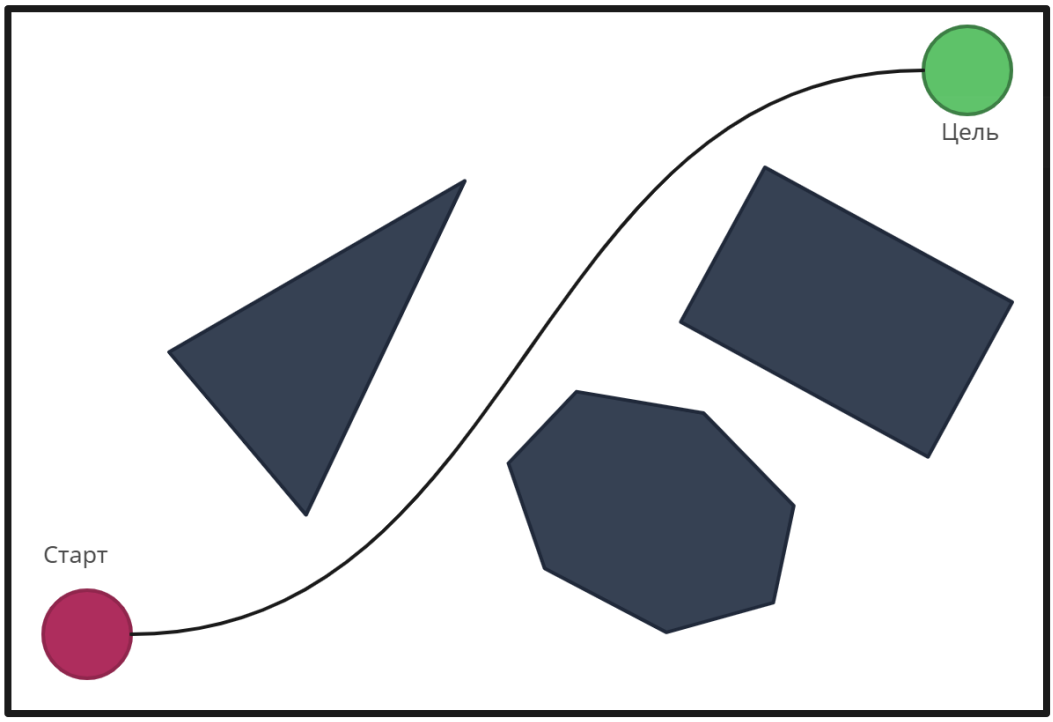

Рисунок 2 – Пример задачи планирования перемещения.

1.2 Конфигурационные пространства

Конфигурационное пространство является набором параметров однозначно определяющее положение робота в пространстве, которые символизирует степени свободы мобильного робота и определяющее конфигурационное пространство объекта. При решении задачи планирования движения, переход необходимо осуществлять к тому пространству, которое будет максимально приближено к искомому пространству.

Число измерений в конфигурационном пространстве равно n числу степеней свободы рассматриваемого объекта. Конфигурационное пространство вводится для представления системы, как некой точки, движимой в данном пространстве [9].

У систем с 1, 2 и 3 степенями свободы (например, у плоского математического маятника, у сферического маятника и у свободной материальной точки) конфигурационное пространство будут соответственно прямая, плоскость и 3-мерное пространство; у свободного твёрдого тела,

имеющего 5 степеней свободы, конфигурационное пространство будет 5-и мерным и т. д. [1].

На рисунке 3 приведён пример манипулятора с 6 степенями свободы захватного устройства. Конфигурационное пространство такого манипулятора является 6-ти мерным.

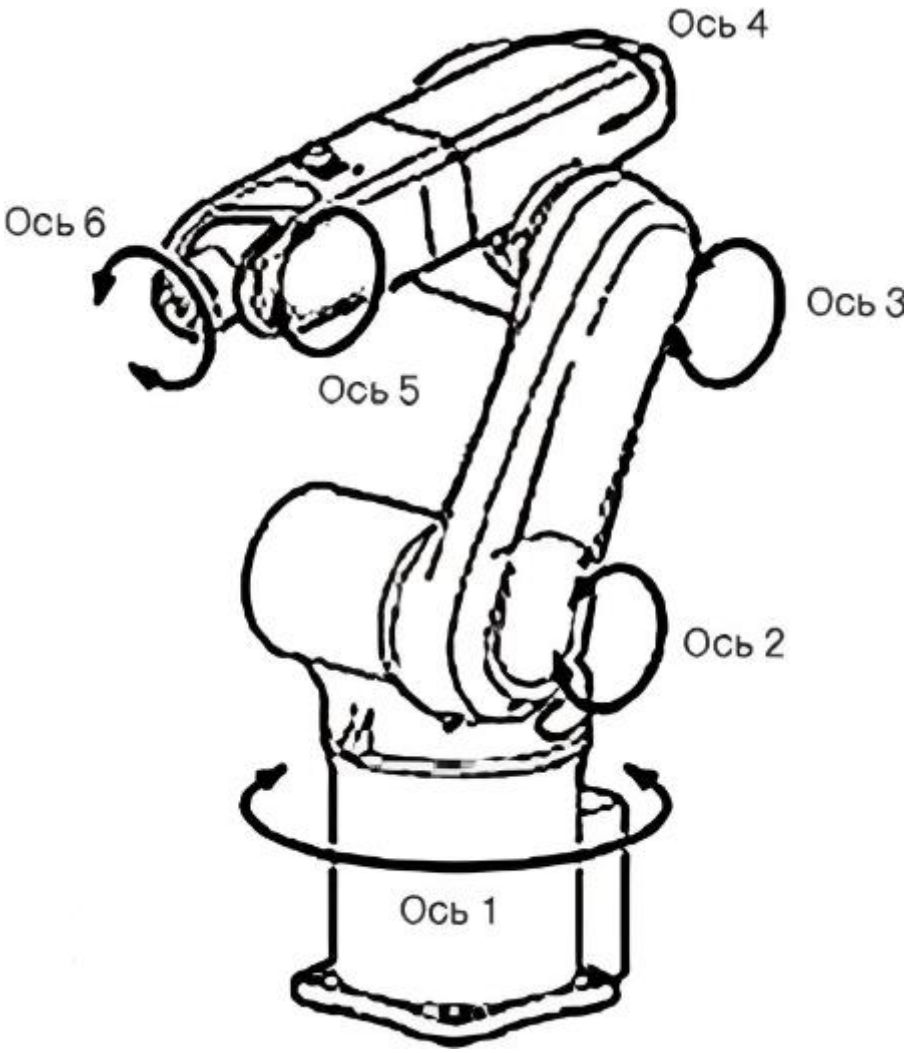

Рисунок 3 – Робот-манипулятор с 6 степенями свободы.

Переход к конфигурационному пространству предоставляет возможность описывать положения робота, а также возможные перемещения с повышенной точностью. Каждая точка этого пространства является возможной конфигурацией робота, поэтому задача сводится к задаче поиска пути из начальной к конечной конфигурации. Такой подход учитывает поведение робота

в окружающей среде и позволяет решать более сложные задачи планирования пути, такие как задачи с динамическими препятствиями, учёт скорости и ускорения робота, а также увеличивает точность и эффективность в расчёте траектории мобильных роботов в решении задач планирования движения.

Движение системы однозначно определяет своё положение в течении всего времени, поэтому в любой момент можно определить её конфигурацию.

Множество конфигураций, удовлетворяющее всем кинематическим ограничения и избегающее конфликта с объектами препятствий называется пространством допустимых состояний.

В качестве примера на рисунках 4-6 представлен переход к конфигурационному пространству робота, представленного в виде многоугольника, и объекта препятствия в виде прямоугольника. Для учётов всех рисков столкновения, робот представляется точкой, а препятствие увеличивается по ширине и высоте на половину габаритов робота.

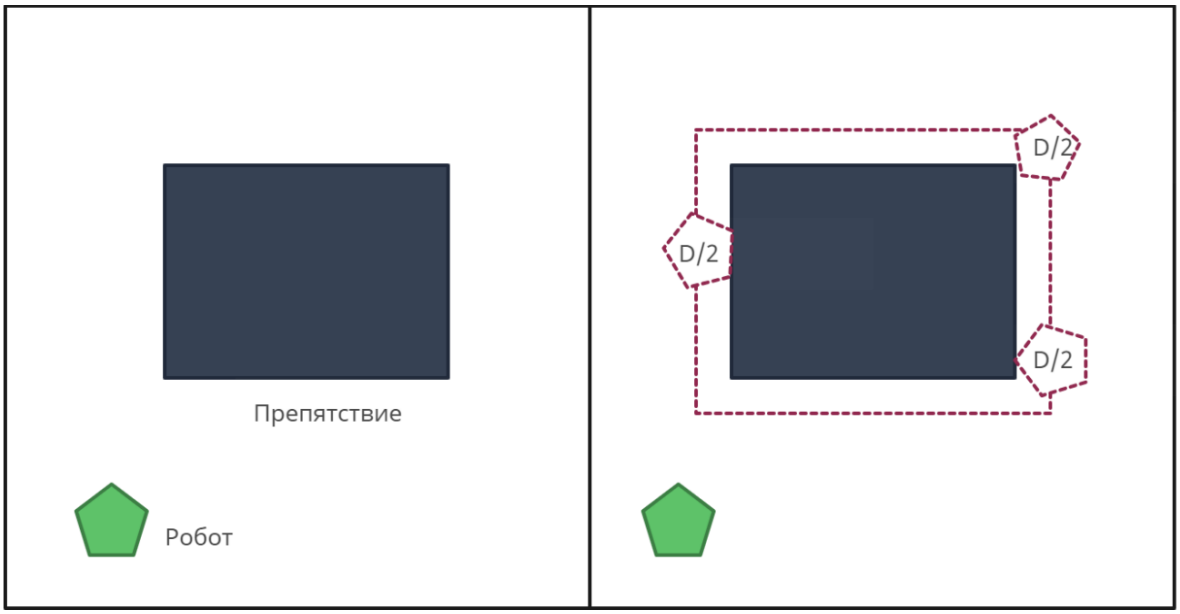

Рисунок 4-5 – Увеличение препятствия по длине и ширине на максимальное измерение робота.

Далее, на рисунке 6 показано конфигурационное пространство, где робот является точкой, а препятствие увеличилось по длине и ширине на величину равной половине ширины робота.

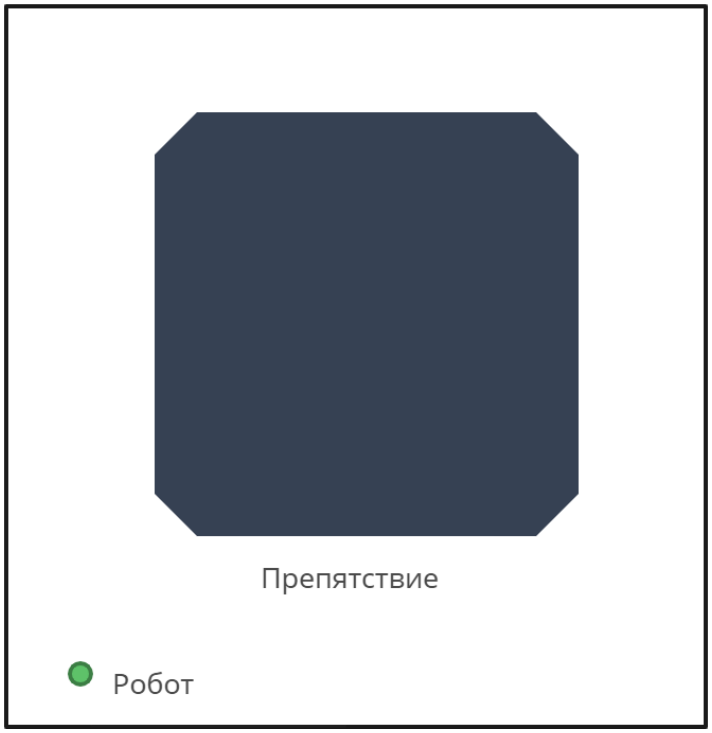

Рисунок 6 – Конфигурационное пространство робота с препятствием.

На рисунке 7 показан пример перехода робота в пространство конфигураций, где препятствие, представленное окружностью, перемещается во времени.

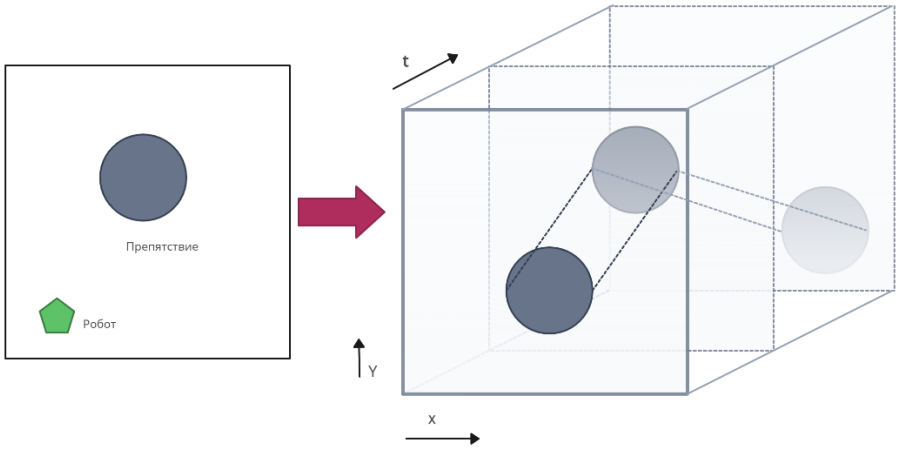

Рисунок 8 – Переход робота в трёхмерное конфигурационное пространство

На рисунке 9 представлено кинематическое представление системы с 5 степенями свободы.

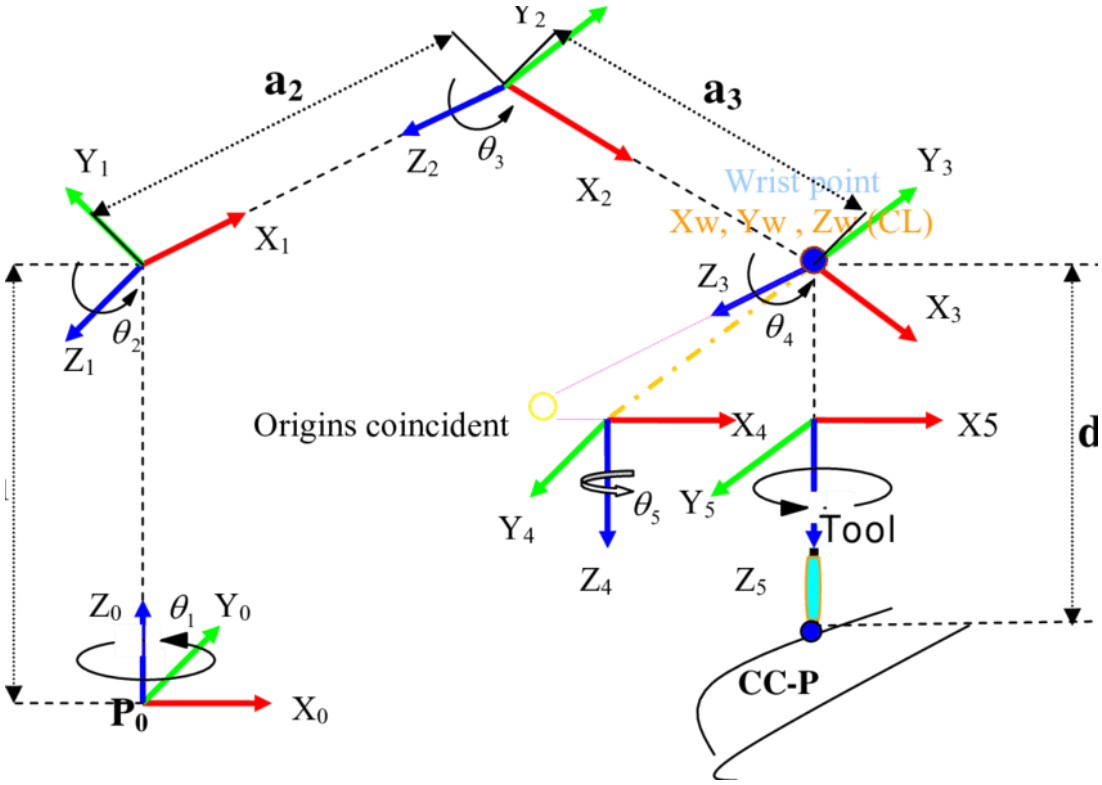

Рисунок 9 – кинематическое представление системы с 5 степенями свободы

При реализации алгоритма МИПП в данной дипломной работе будет использовать приложение генератора препятствий, которое создаёт препятствия в двумерном пространстве, поэтому рассматриваемое конфигурационное пространство будет соответствующим.

2 Способы решения задачи планирования движения

2.1 Классификация и обзор методов планирования движения

Методы планирования движения классифицируются по разным признакам. В контексте использования интеллектуальных технологий их можно разделить на точные и эвристические. По характеру окружающей обстановки можно разделить методы планирования на методы планирования в статической окружающей среде и в динамической среде. Методы также можно разделить по полноте информации об окружающей среде: методы с полной информацией и методы с неполной информацией знании обстановки в непосредственной близости от робота, в этом случае речь идет о локальном планировании пути [3].

Принято различать способы планирования движения на следующие категории:

1) Оптимизационные методы;
2) На основе графов;
3) На основе клеточной декомпозиции;
4) На основе интеллектуальных технологий;
5) Метод иерархии подцели;
6) Метод потенциальных полей;
7) Вероятностные методы

Каждая из категорий методов имеет большое разнообразие алгоритмов и модификаций [15].

2.1.1 Оптимизационные методы

Задачу планирования пути можно решить, как оптимизационную задачу. Для этого необходимо представить движение объекта в рамках какой-либо

динамической системы. Препятствия описываются некоторыми ограничениями, а качество допустимой траектории должно оцениваться некоторым функционалом. В результате возникает задача оптимального управления, которая позволяет выбрать лучший вариант по скорости прохождения и обеспечивает обход препятствий при построении траектории объекта [10].

2.1.2 Методы на основе графов

Методы на основе графов являются достаточно популярными среди исследователей в последнее время. Основные методы этого класса показаны на рисунке 10.

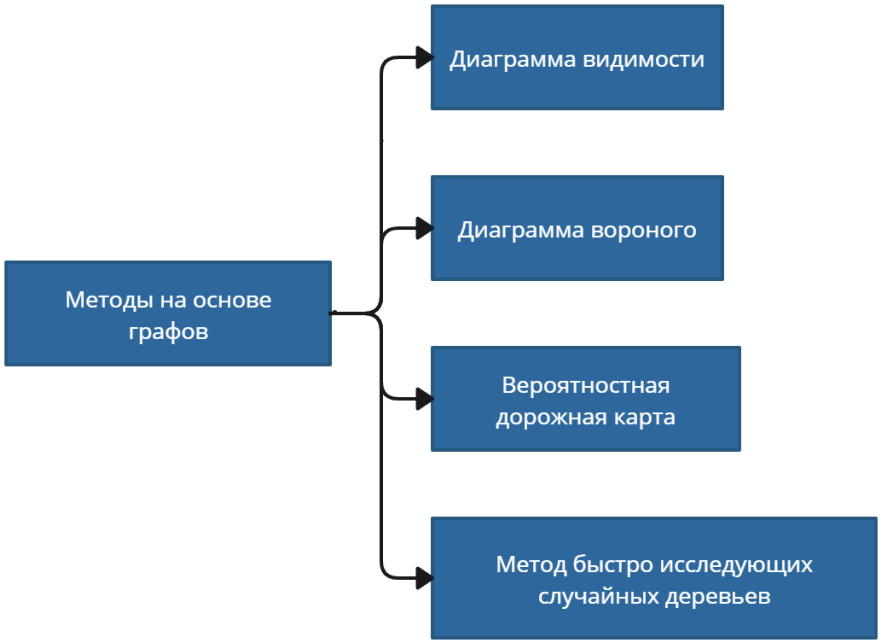

Рисунок 10 – Методы, основанные на графах.

Граф (или дерево) отражает состояния, в которых может находиться робот, при этом каждый узел соответствует определенному состоянию робота. Состоянием является угол, ориентация, положение робота в пространстве. Переходы между состояниями характеризуются функциями затрат. Это позволяет выделить путь, который имеет минимальную общую стоимость достижения конечного состояния [3].

2.1.3 Методы на основе клеточной декомпозиции

Суть метода заключается в дискретизации окружающей среды. Существует две основные группы клеточной декомпозиции: точная и приближённая (cell decomposion).

При точной клеточной декомпозиции среду, в которой находится объект, делят на клетки, задействуя грани препятствий. В данном методе играет немаловажную роль задача разложения многоугольника на выпуклые области. Для этого используются такие решения как вертикальное и трапециевидное разложение [14].

Приближённая клеточная декомпозиция осуществляется с помощью сетки, покрывающей пространство (grid map). Идея метода заключается в разделении пространства окружающей среды на клетки одинакового размера. Каждой клетке присваивается значение 1 (препятствие) или 0 (нет препятствия).

Данную сетку очень просто использовать и поддерживать. Из минусов можно выделить увеличение сложности вычислений при уменьшении шагов сетки. Особенно сильно проявляется данный недостаток при окружающей среде большого объёма [3].

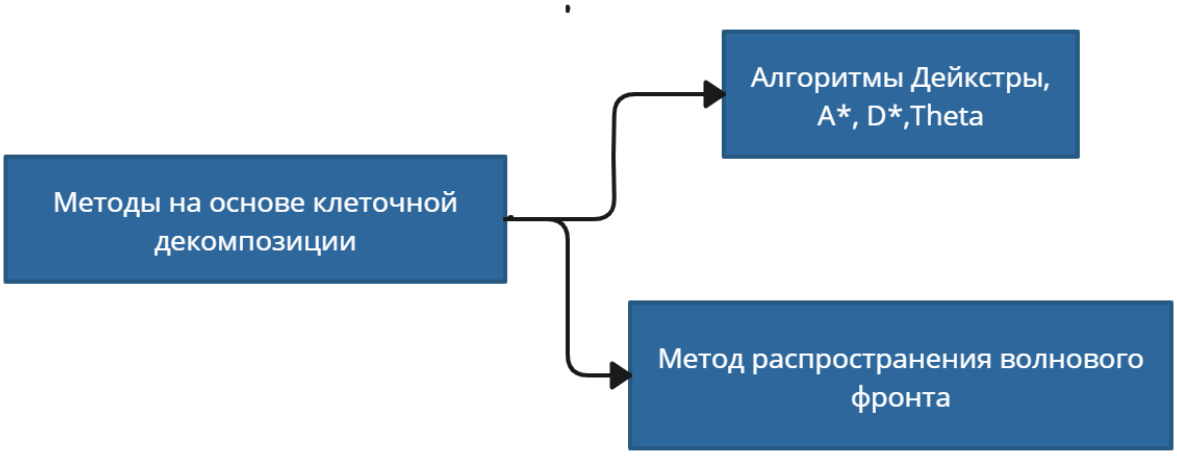

Рисунок 11 – Разновидность методов на основе клеточной декомпозиции.

2.1.4 Методы на основе интеллектуальных технологий

Робот должен уметь решать задачу планирования движения в условиях реальной окружающей среды без помощи человека. В таких ситуациях естественно использовать алгоритмы, основанные на поведенческом факторе, биоспирированные алгоритмы. Данные методы имитирует поведение и мышление человека и других биологических существ.

Некоторые виды поведенческих алгоритмов представлены на рисунке 12.

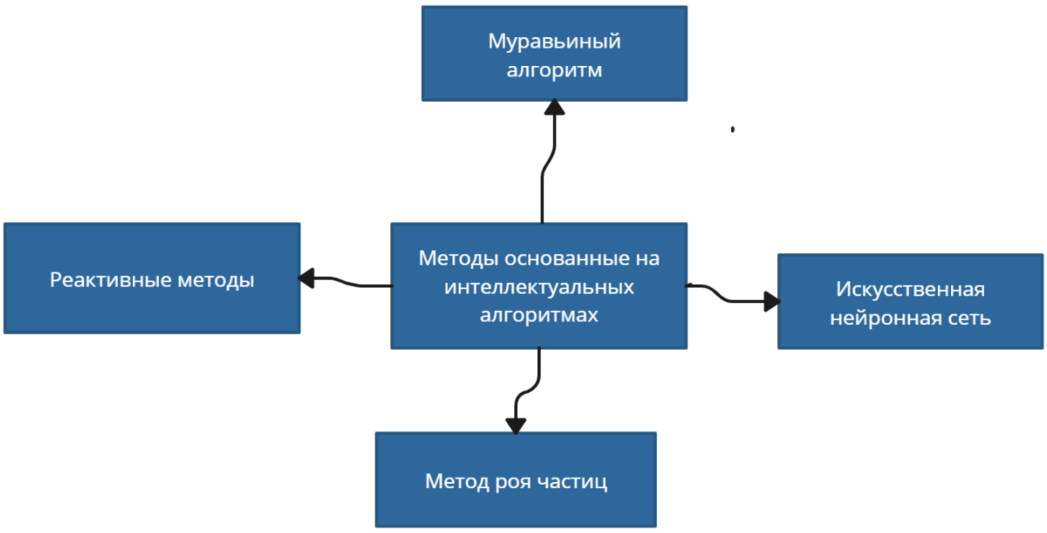

Рисунок 12 – Некоторые разновидности биоспирированных методов.

Муравьиный алгоритм направлен на решение трудоёмких комбинаторных задач. С помощью данного метода ищется близкое решение за оптимальное время.

Искусственная нейронная сеть – это группа математических моделей, имитирующих нервную систему человека. Сеть клеток образуют совокупность однородных элементов, называемых нейронами, каждый из которых имеет один или несколько входов и один или несколько выходов. Входы и выходы соединяют подобие организма в единую систему. Часть входов и выходов являются обобщающими для всей системы. Когда на вход подаётся сигнал, нейронная сеть обрабатывает его и результат выдаёт выходные нейроны. Каждый нейрон обладает несколькими параметрами, регулирующие его работу.

Метод роя частиц имитирует движение стаи птиц в контексте “коллективного интеллекта” движения популяций биологических видов. Данный метод выявляет способность различных популяций адаптироваться к среде обитания, выживанию (избежание препятствий) и обмену информации. Перемещения подчиняются принципу наилучшего найденного положения объекта, которое изменяется в зависимости более выгодного положения.

Реактивные методы являются важной группой алгоритмов, основанных на интеллектуальных технологиях. В основе этих алгоритмов лежат концепции поведения, решающего типовые задачи в типовых ситуациях. В учёт идёт вся масса обстоятельств, возможная при условиях реальной окружающей среды.

2.1.5 Методы иерархии подцели

Методы иерархии подцели руководствуются принципом использования двухуровневой схемы планирования (рисунок 12) без уменьшения требования памяти.

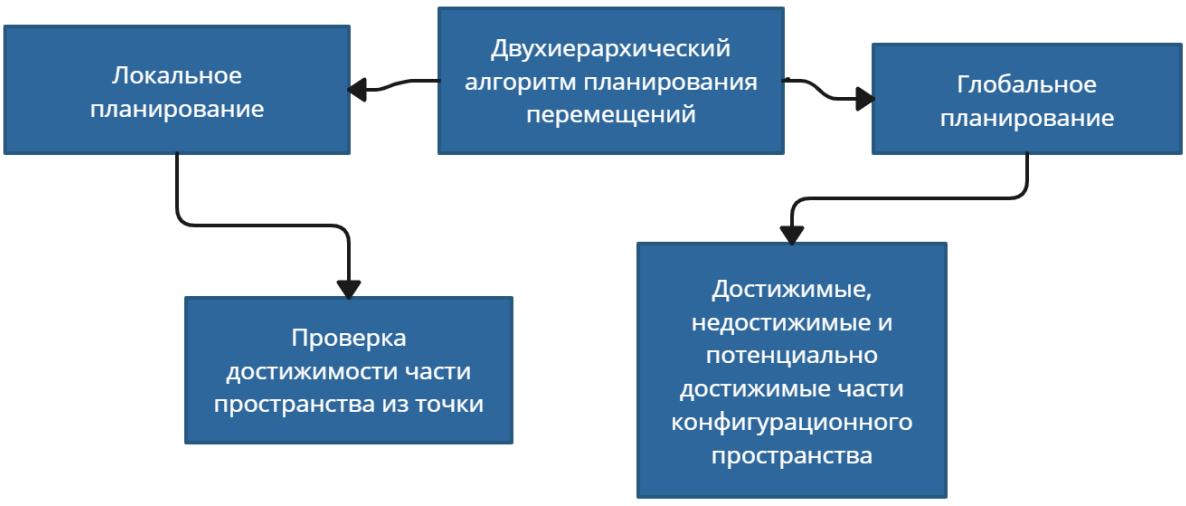

Рисунок 13 – Принцип работы двухуровневого алгоритма.

Примером алгоритма иерархического метода является алгоритм SANDROS. Алгоритм САНДРОС, способен планировать движения для

произвольных роботов, работающих в средах с препятствиями. Движение робота планируется в конфигурационном пространстве. Следующие проектные решения имеют решающее значение в разработке эффективного планировщика движения без ущерба для полноты разрешения. В алгоритме используется алгоритм расстояния для вычисления свободных от столкновений частей конфигурационного пространства. Можно использовать условия взаимодействия робота с препятствиями, чтобы узнать количество шарниров, свободных от столкновений. Однако информация о расстоянии между роботом и препятствиями в конкретной конфигурации позволяет нам выбрать более “безопасные” для робота [1].

2.1.6 Вероятностные методы

Для проектирования перемещения в сложных и забитых препятствиями сценах применение методов, основанных на точных или приближенных восстановлении пространства допустимых позиций, часто является неоправданно сложным. В таких случаях используются вероятностные методы, которые позволяют исследовать области пространства позиций путем многочисленных экспериментов. Таким образом, можно получить результаты о возможности выбранных отдельных позиций. На основе этих результатов создается маршрутная сеть, что является преимуществом этого подхода - независимым от геометрического представления и размерности моделируемой среды [11].

Среди вероятностных методов можно выделить методы:

1) Построения дерева всевозможных путей;
2) Нахождение наиболее вероятной траектории;
3) Алгоритм определения траектории;
4) Дискретизация пространства поиска;
5) Алгоритм распознавания цветов.

2.2 Метод искусственных потенциальных полей

Метод потенциалов в задаче выбора пути для мобильного робота (МР) был предложен А.К. Платоновым в 1970 году.

Рассматривается система с достаточно точной навигационной системой ошибками которой можно было пренебречь. Системе управления известны как координаты робота и измерительного устройства, так и ориентация сектора обзора и направление производящихся измерений в некоторой абсолютной системе координат (АСК). Робот во всех случаях представляет собой точку с предписанным вектором ориентации [12].

Основная идея МИПП заключается в исследовании пространства в процессе поиска и оценка областей пространства по потенциалу, а не по контакту с препятствием. Потенциал – показатель перспективности и доступности области. В отличие от векторных алгоритмов и др., данный метод может использоваться для пространств большой размерности [7].

Функция $U : R^m \to R$, в которой каждой точке многомерного пространства соответствует число, которое является так называемым потенциалом или энергией точки.

Градиент в каждой точке рассматриваемого пространства является силой, которая направлена в сторону максимального возрастания U. То есть каждой точке сопоставляется вектор, что превращает конфигурационное пространство в векторное поле.

Робот рассматривается как положительно заряженная точка, которая отталкивается от препятствия и притягивается к отрицательно заряженной цели, которая является обычно концом маршрута движения робота. В каждой точке потенциального поля на робота действует комбинация сил притягивания и отталкивания, которая в итоге должна направить робота к цели, избегая препятствия.

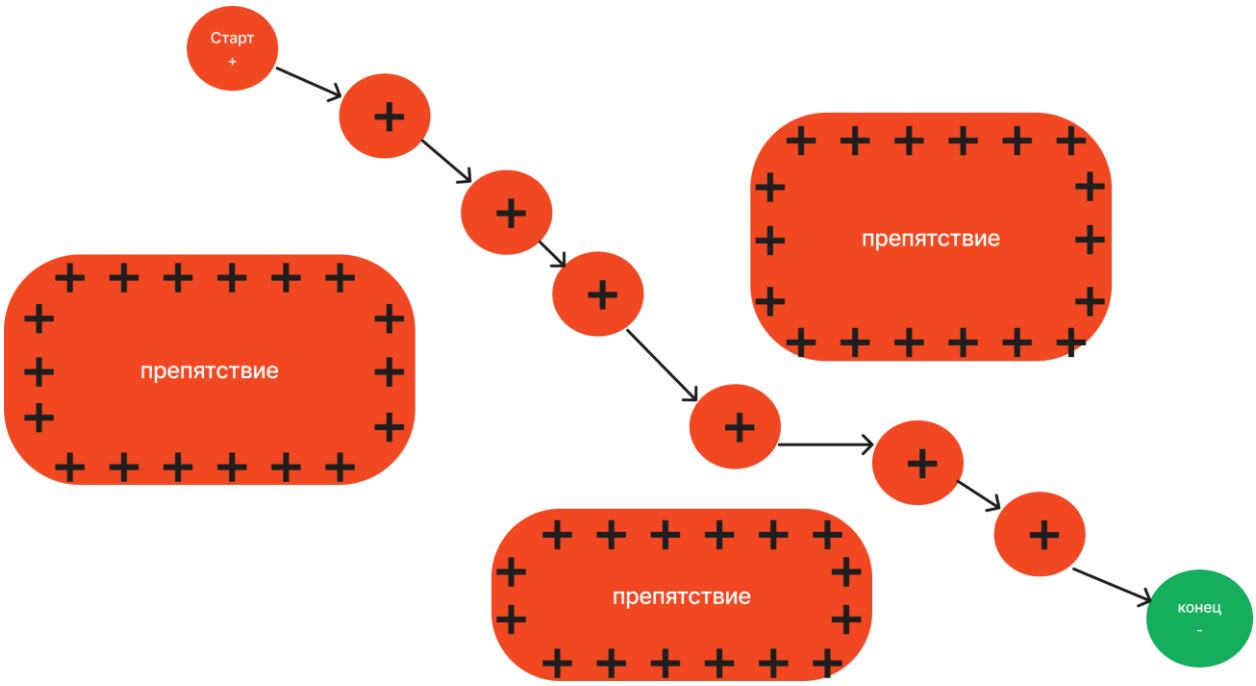

Рисунок 14 – Движение точки в потенциальном пространстве

При расчете маршрута считается, что навигация робота обладает точностью, при котором погрешность движения считается близкой к нулю, в любом случае робот представлен вектором. Однако в реальных условиях роботы могут обладать меньшей точности, поэтому в различных ситуациях возможно рассчитать степень “риска”, под которым понимается близость прохождения робота с препятствиями. Пример работы алгоритма представлен на рисунке 14.

2.3 Описание работы алгоритм МИПП

Данный алгоритм рассматривает препятствия с точки зрения объектов, создающих силу отталкивания. Траектория движения мобильного робота является кривой, поэтому для получения точек перемещения, кривую необходимо аппроксимировать. Аппроксимация проводится путём дискретизации конфигурационного пространства. Шаг дискретизации определяется эвристическим методом, в результате оценки плотности и размера препятствий. В результате ряда тестирований было установлено, что шаг дискретизации зависит от плотности и площади, как:

$$D_s = S_{obs} * 2\sqrt{P_{dens}},$$

Где S_{obs} – площадь препятствий, P_{dens} – плотность.

Примеры дискретизации представлены на рисунках 15-18.

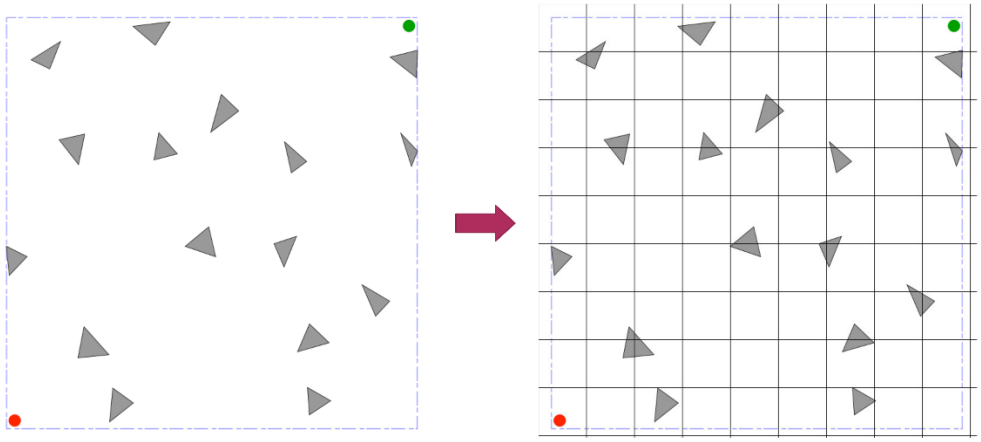

Рисунок 15 – Дискретизация поля с объектами средней плотности и малого размера.

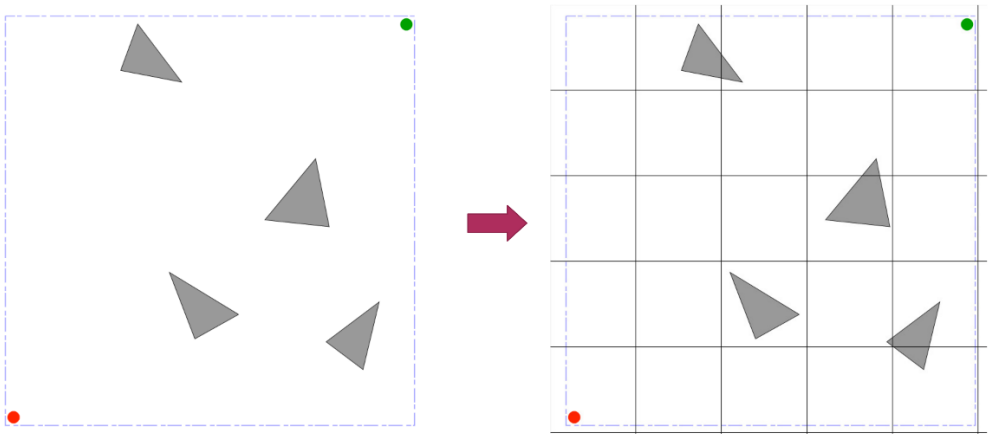

Рисунок 16 – Дискретизация поля с объектами малой плотности и средних размеров.

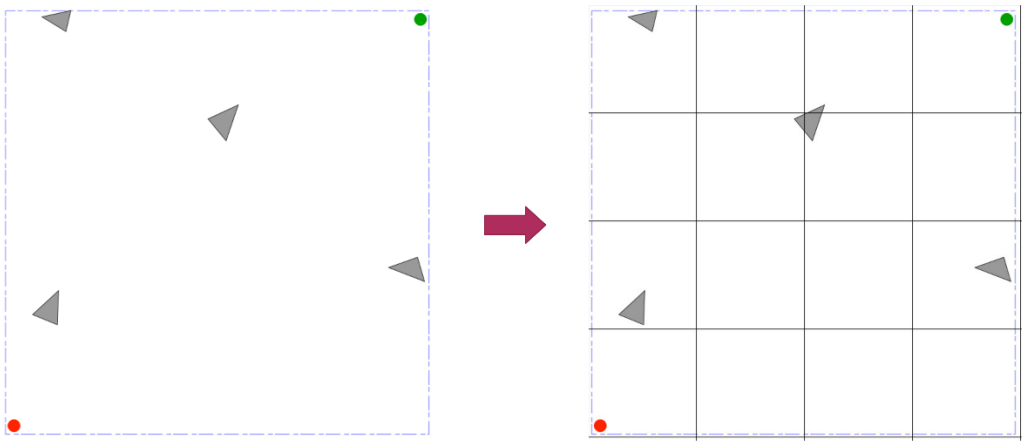

Рисунок 17 – Дискретизация поля с объектами малой плотности и малых размеров.

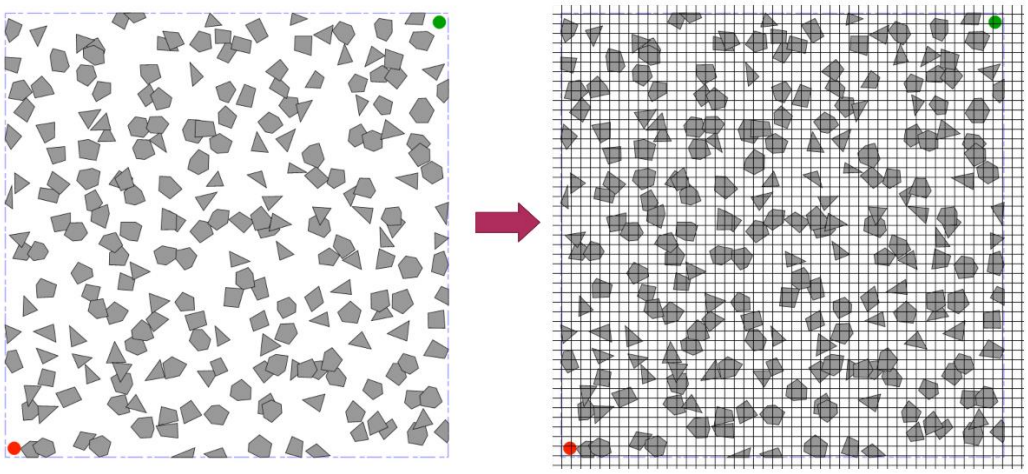

Рисунок 18 – Дискретизация поля с объектами большой плотности и малых размеров

Далее, реализуется процесс заполнения ячеек. Для расчёта комбинированного потенциала, для каждой ячейки рассчитывается расстояние до всех препятствий и вычисление суммарного потенциала. К результату прибавляется сила притягивания ячейки к целевой точке. Для каждой ячейки находится сила отталкивания от каждого препятствия и суммируется с силой притяжения к конечной точке равной евклидову расстоянию. Полученные

величины являются силами, которые действуют на мобильного робота, находящегося в одной из ячеек [7].

Так как задача планирования перемещений подразумевает собой нахождение минимального маршрута среди всех возможных, инициализируется процедура поиска локального минимума, то есть поиск точки, в которой значение потенциала минимально. Существует множество способов нахождения локального минимума. Для этого удобнее всего построить дерево решений методом BFS (Best First Search).

2.4 Потенциалы отталкивания и притяжения

Потенциал в каждой точке считается по формуле
$U(c) = U_{atr}(c) + \sum_i U_{repi}(c)$, где $U_{atr}(c)$ – потенциал притяжения,
$\sum_i U_{repi}(c)$ – потенциал отталкивания от i-го препятствия [7].

В данной работе $U_{repi}(c) = (\frac{1}{D})^2$, где D – расстояние для препятствия.

Суммирование происходит по всем препятствиям в потенциальном поле. Потенциальная функция монотонно возрастает по мере отдалении от цели.

Потенциал отталкивания выполняет функцию отталкивания от препятствий. Чем дальше робот от цели, тем быстрее он приближается к ней и наоборот. Функция должна возрастать по мере приближения робота к непроходимой зоне.

Сила притяжения представляет собой функцию, увеличивающую своё значение по мере приближения робота к целевой точке. Рассчитывается по формулам Манхеттенского:

$$U_{atr} = (c - t),$$

где c – целевая точка, t – текущее положение рассматриваемого объекта, равна сумме разности их координат [13].

Или по формуле Евклидова расстояния:

$$U_{atr} = \sqrt{(x_c - x_t)^2 + (y_c - y_t)^2}$$

Общий градиент передвижения можно найти по формуле

$$f_{all}(c) = -\nabla U_{all}(c) = -\nabla \sum_i U_{all}(c) = \sum_i f_i$$

Для реализации расчёта расстояния до препятствия, можно воспользоваться различными методами нахождения расстояния между точкой и геометрическими фигурами. На начальном этапе исследований в ИПМ рассматривались препятствия в виде окружностей. Данный метод имеет достаточно низкую точность из-за проблемы существования вытянутых фигур. Пример данной проблемы представлен на рисунке 19.

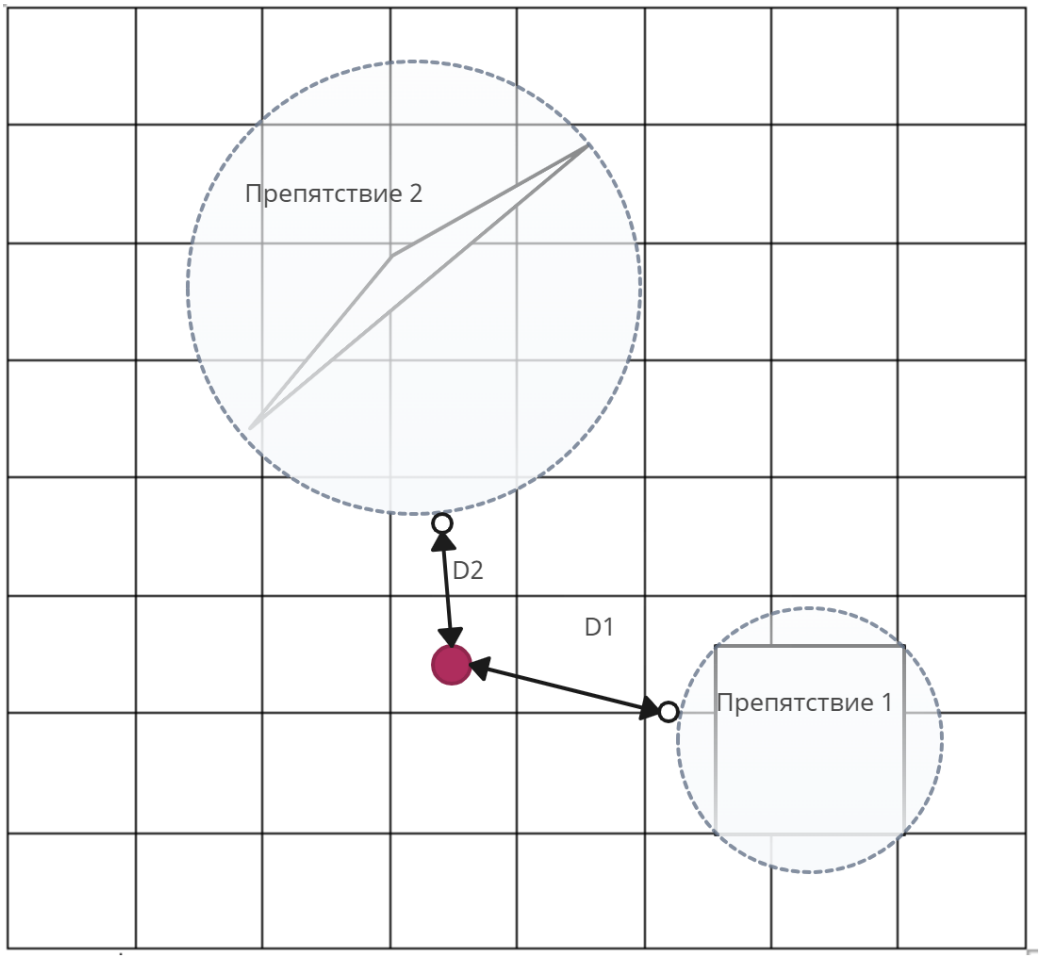

Рисунок 19 – Расстояние, вычисленное методом построения окружности, с примером вытянутой фигуры показывает низкую точность, так как, большая часть фигуры, через которую потенциально мог пройти мобильный робот, принимается за препятствие.

В данной работе используется метод расчёта расстояния между точкой и ближайшей точкой препятствия.

Данный метод гарантирует обход препятствий и обладает довольно высокой вычислительной точностью.

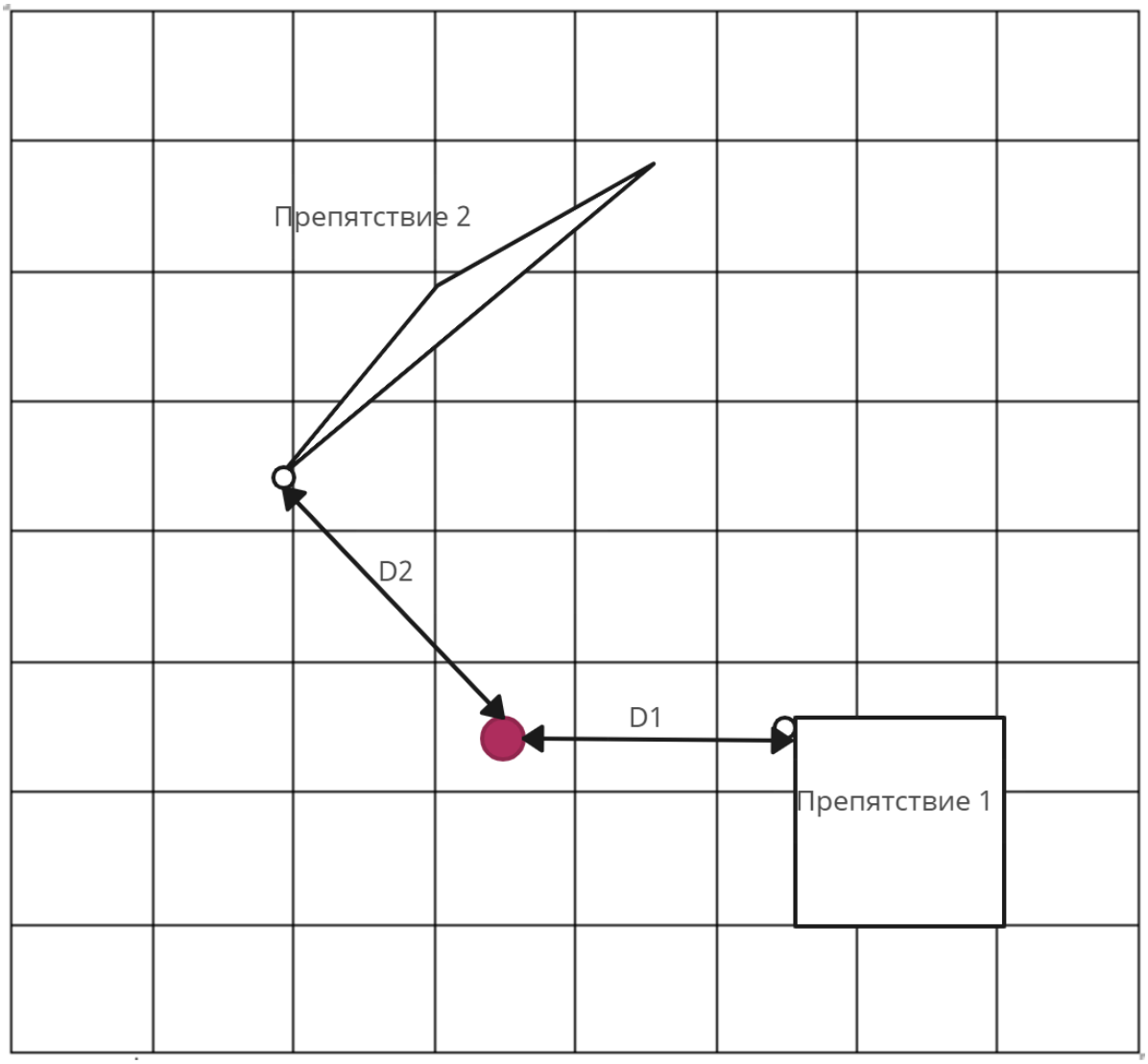

Рисунок 20 – Расчёт расстояния методом ближайшей точки.

Как видно из рисунка 20, метод расстояния ближайшей точки находит дистанцию до объекта гораздо точнее, чем методом окружности. В дальнейшем будет рассматриваться именно этот способ.

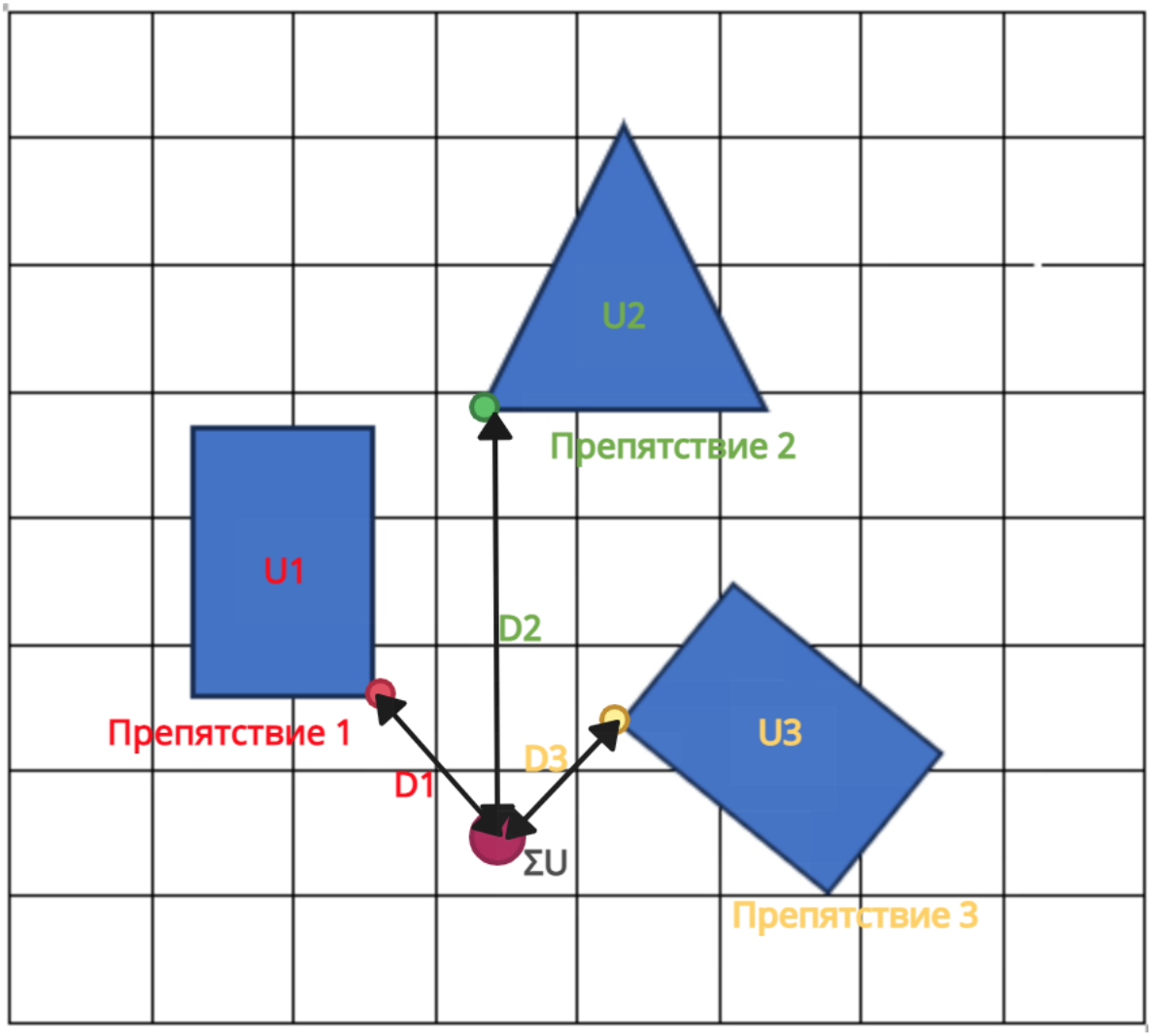

Рисунок 21 – Вычисление потенциала для рассматриваемой ячейки

На рисунке 21 приводится пример вычисления потенциала для поля размером 8 на 8.

Находится расстояние до всех препятствий, далее по формуле $\sum_i U_{repi}(c) = \sum_i (\frac{1}{D_i})^2$ находится потенциал отталкивания данной ячейки.

Для расчета оптимальной комбинированной функции отталкивания и притяжении требуется грамотно определять коэффициенты в градиентном спуске или же алгоритме Best first search. Сила притяжения, например основанная на Манхэтеннском или Евклидовом расстоянии [13,14], должна соотноситься с силой отталкивания так, что силы отталкивания будет достаточно, чтобы исключить риск при прохождении препятствий и

одновременно обеспечить проходимость между препятствий, расстояние между которыми допускается при выбранном шаге алгоритма. Для этого необходимо подобрать коэффициенты уравнений потенциала эвристическим методом.

2.5 Метод поиска локального минимума BFS

Алгоритм поиска наилучшего является одним из самых простых способов поиска локального минимума. Предполагается, что свободная область конфигурационного пространства дискретизирована и для каждой ячейки получен потенциал [7].

1. В процессе решения создаётся дерево поиска T, которое на первом шаге состоит из одной стартовой вершины;
2. Среди листьев выбирается значение с минимальным значениям потенциала U;
3. Ищутся все соседи S(p);
4. Эти ячейки становятся потомками вершины p в дереве поиска T;
5. Поиск продолжается, пока вершина t не попадёт в окрестности какого-нибудь из листьев.

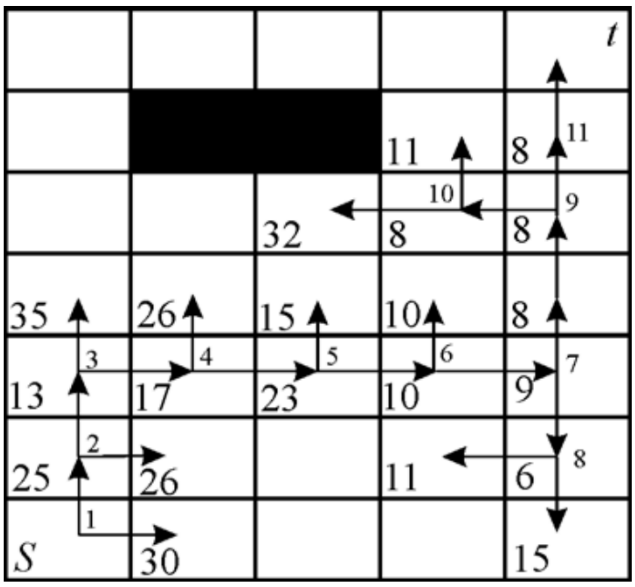

Рисунок 22 – пример построение дерева решений методом BFS.

3 Программная реализация

3.1 Реализация метода МИИП

Для реализации МИИП было принято решение воспользоваться функционалом языка PYTHON.

Преимущество данного языка заключается в возможности использования библиотек, содержащие математические алгоритмы, что значительно упрощает выполнение поставленной задачи.

Были созданы классы Field и Cell. Класс Field содержит в себе методы необходимые для реализации искусственного потенциального поля.

Описание методов и полей класса представлено в таблицах 1,2.

Таблица 1 – Описание полей класса Field.

Название поля	Использование
size	Инициализация стандартного размера поля
start, end	Инициализация функций преобразования координат
iters	Ограничение по количеству итераций
step	Шаг дискретизации
field	Заполнение поля нулями с помощью функции np.zeros(size,size)
_fill_field	Создание поля с объектами класса Cell с помощью функции np.zeros(size,size)
beta	Объявление константы коэффицента при рассчёте силы отталкивания от препятствий

Таблица 2 – Описание методов класса Field

Название метода	Описание назначения метода
def __init__(self, size=100, iters=2000, start=(2, 2), end=(98, 98))	Констуктор класса, принимает параметры, необходимые для решения алгоритма, инициализирует размерность, шаг
def _prepare_start_end(self, obj)	Преобразование абсолютных координат в локальные с учётом шага дискретизации
def _fill_field(self)	Заполняет поле объектами класса Cell
def _stepping(self, x)	Получение числа кратного шагу. Используется для сохранения точности вычислений
def get_barriers(self)	Расположение препятствий на поле
def _get_neighbours(self, cell)	Получение соседей клетки
def _get_beta(self)	Нахождение beta для расчёта силы отталкивания
def gradient_descent(self)	Функция нахождения потенциала в каждой точке поля
def _find_min(arr)	Нахождение точки с минимальным потенциалом из списка
def best_first_search(self)	Реализация алгоритма BFS
def _find_min_dist(self, leaves)	Нахождение точки с минимальным расстоянием до конечной точки
def _create_way(self, tops, is_end)	Преобразование в json файл массива точек
def show_2d_capability(self, is_d=False):	Создание 2д карты
def show_3d_capability(self)	Создание 3д карты
def _normalize(field):	Нормализация данных для отображения на графике

В функции main() происходит инициализация функций создания поля и поиска кратчайшего пути в алгоритме.

Сначала инициализируется класс Field, задаются начальные атрибуты класса, передаётся начальная и конечная точка, количество итераций. Для нахождения зависимости шага дискретизации от плотности объектов была реализована функция get_size(), рассчитывающая примерную площадь препятствий поля 100 на 100. Создаётся словарь списков областей каждого препятствия, при попадании препятствия в ячейку, в неё соответственно заносится значение число 1, в противном случае заносится – 0. Идея зависимости в том, что в зависимости от процента занятости поля и плотностью препятствий, необходимо установить фиксированные значения шага.

По формуле $D_s = S_{obs} * 2\sqrt{P_{dens}}$ находится шаг сетки.

Далее в файле main() вызывается функция get_barriers(), отвечающая за расстановку препятствий на поле. Для каждого препятствия из списка barriers начиная с 4 элемента (так как первые три элемента передают начальную, конечную точку и характеристики поля препятствий) создаётся список вершин фигуры points_list. Список заполняется данными типа кортеж, в котором хранятся координаты xy. Координаты препятствия добавляются в словарь list_of_ones с ключом равным номеру препятствия. Далее проходим по всем парам вершин в гранях, все ячейки, пересечённые границами периметра препятствия заполняются в словарь list_of_ones. Далее идёт заполнение внутренностей фигур следующим образом: если среди yx в list_of_ones на одной линии всего две координаты, проходим от минимальной в строке до максимальной с шагом равной ячейке и добавляем соответствующую ячейку в словарь list_of_ones.

Наносим все фигуры препятствий на поле, для этого вызываем атрибут is_polygon класса cell и присваиваем ему значение 1.

После добавление препятствий в main() вызывается функция find.distance(), отвечающая для нахождения всех расстояний до препятствий для

каждой ячейки. Для этого для каждой ячейки проверяем принадлежность к списку препятствий, находим расстояние до ближайших точек препятствий.

Далее вызываем функцию gradients_decent, рассчитывающую потенциалы для каждой ячейки поля. При расчёте потенциала отталкивания используется списки расстояний, найденные в предыдущем пункте. Сила притяжения рассчитывается как евклидово расстояние до целевой точки.

После расчёта потенциалов, инициализируется функция best_first_search(), отвечающая за построение дерева решений и поиска минимального маршрута на нём. Для реализации алгоритма использовалась функция Tree(), библиотеки treelib. С помощью возможностей данной библиотеки был реализован алгоритм построения дерева. Суть алгоритма – в поиске минимального текущего маршрута. При каждой итерации дерево ищет новый путь на графе, при обнаружении такового, из конечной точки найденного маршрута образуются “листья” в соседние точки, данная точка становится для них родительской. Итерации повторяются до достижения терминальной точки t, равная 0 и означающая конец пути, или до ограничения по количеству итераций.

3.2 Тестирование работы программы

Для тестирования эффективности и работоспособности алгоритма, необходимо провести тесты с различной плотностью, величиной и размером препятствий.

Для начала проверим работу алгоритма на параметрах:

Таблица 3

Плотность	Количество вершин	Размер
1	3	1

При заданных параметрах площадь, занимаемой фигурой равняется 3. По заданной эвристике шаг дискретизации равняется 20.

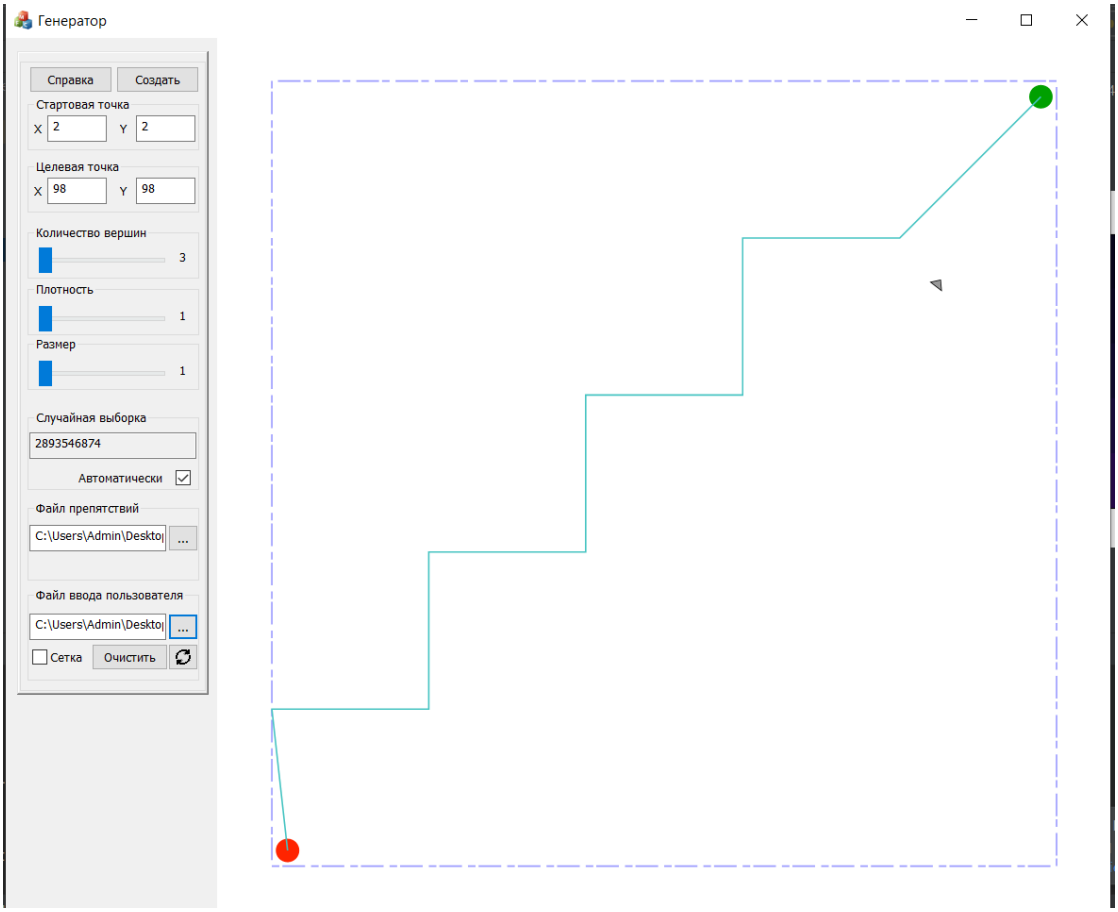

Рисунок 23 – Результат работы алгоритма при шаге дискретизации равному 20.

Таблица 4

Шаг дискретизации	Время заполнения поля	Время работы BFS
20	0.0000001 сек	0.01 сек

Для анализа значения плотности в выборе шага интегрирования проведём эксперимент, задав одинаковую площадь при различной плотности препятствий.

Таблица 5 – Тест 1

Плотность	Количество вершин	Размер
6	3	3

Таблица 6 – Тест 2

Плотность	Количество вершин	Размер
2	3	11

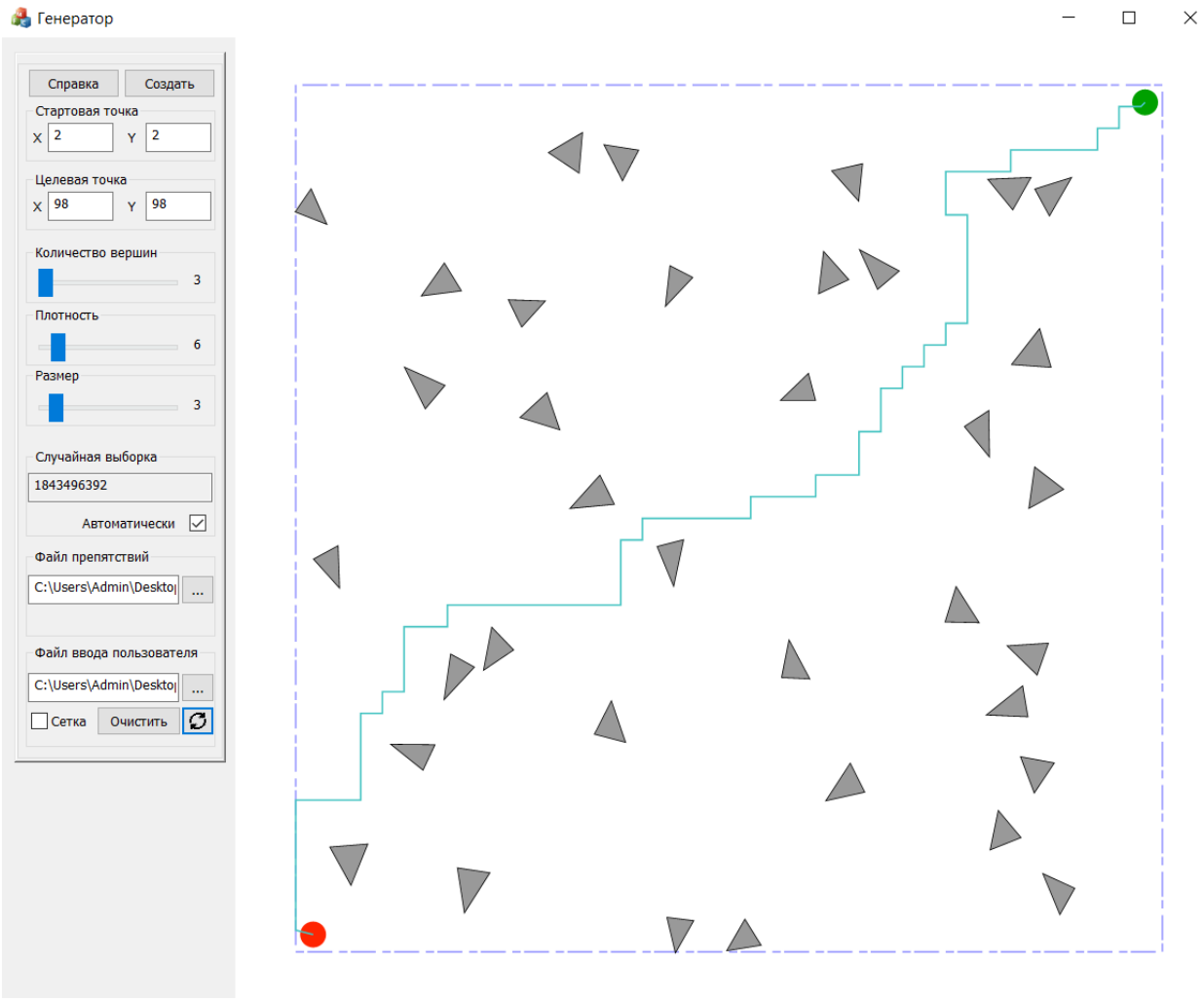

Рисунок 23 – результат теста 1.

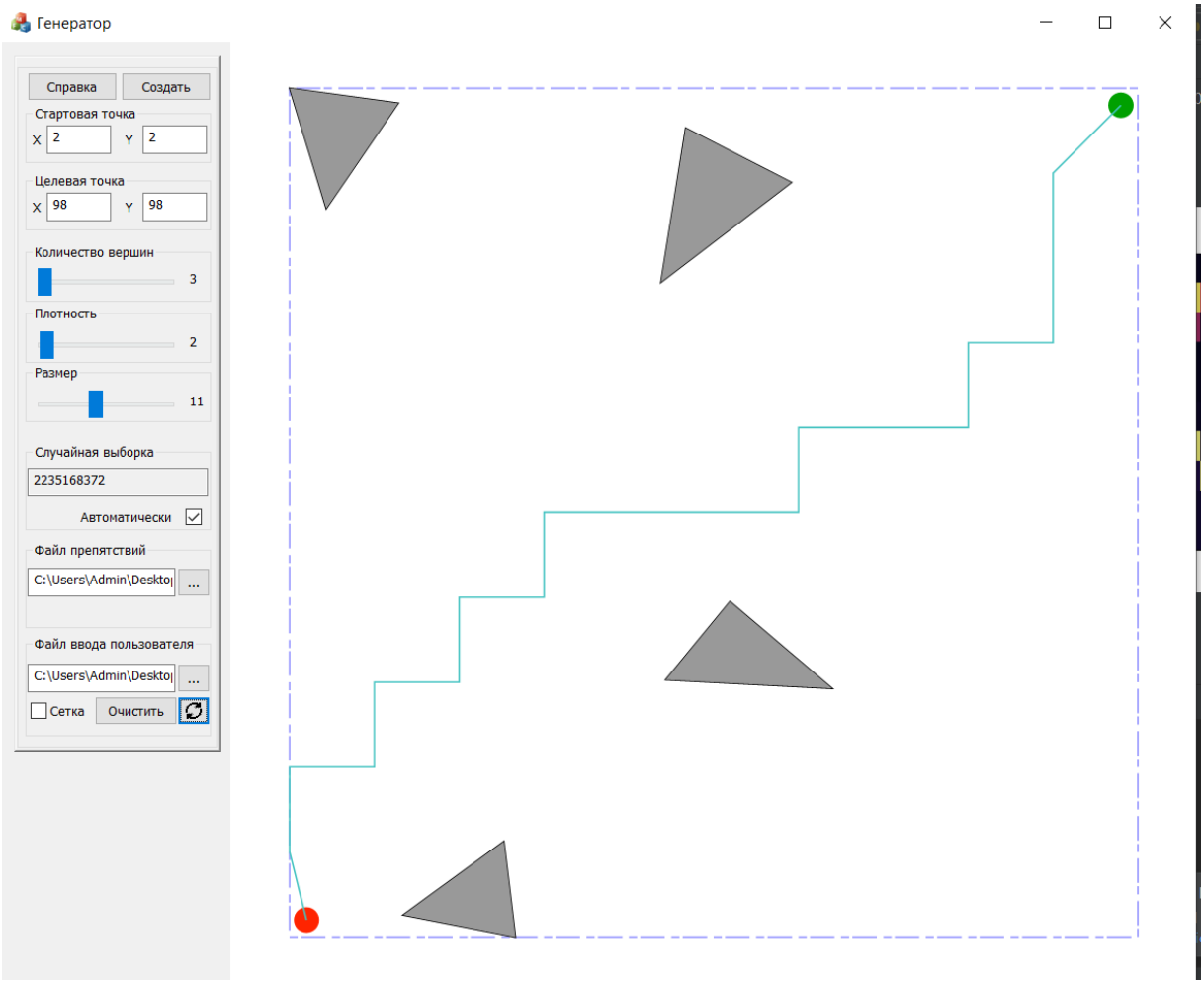

Рисунок 24 – результат теста 2.

Таблица 7 – Сравнение результатов

	Шаг дискретизации	Время заполнения поля	Время работы BFS	Площадь
Тест 1	2.5	3.56 сек	0.01 сек	465
Тест 2	10	0.02 сек	0.03 сек	434

В результате эксперимента была установлена зависимость шага дискретизации от плотности при одинаковой площади.

Усложним тесты, повысив плотность и размер на несколько пунктов.

Таблица 8 – Тест 1

Плотность	Количество вершин	Размер
6	3	3

Таблица 9 – Тест 2

Плотность	Количество вершин	Размер
2	3	11

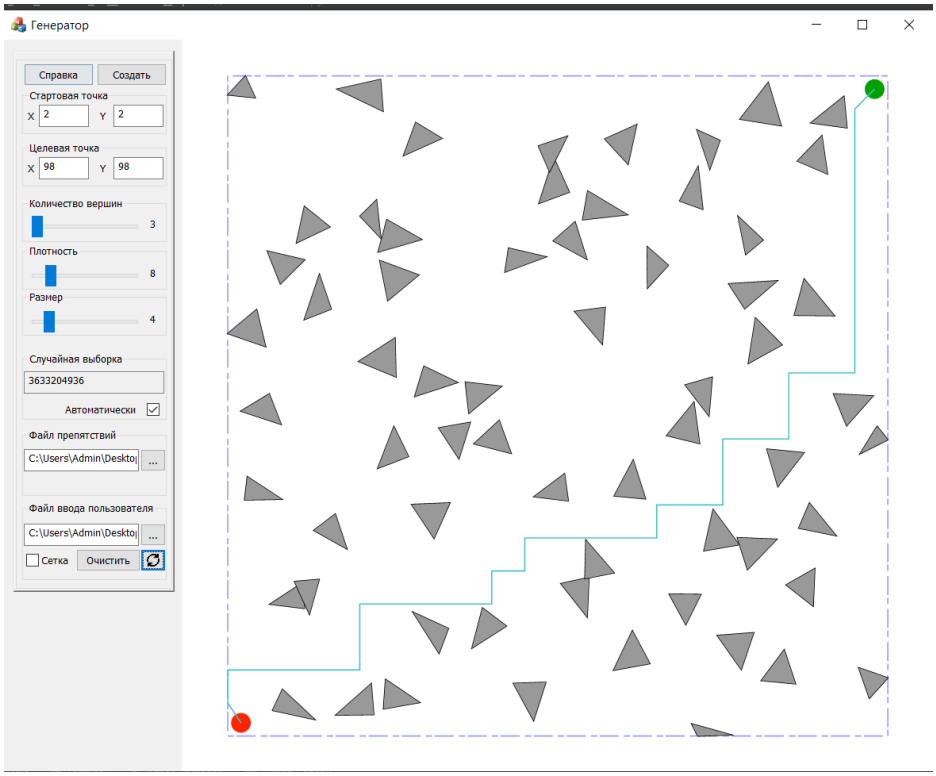

Рисунок 25 – результат теста 1.

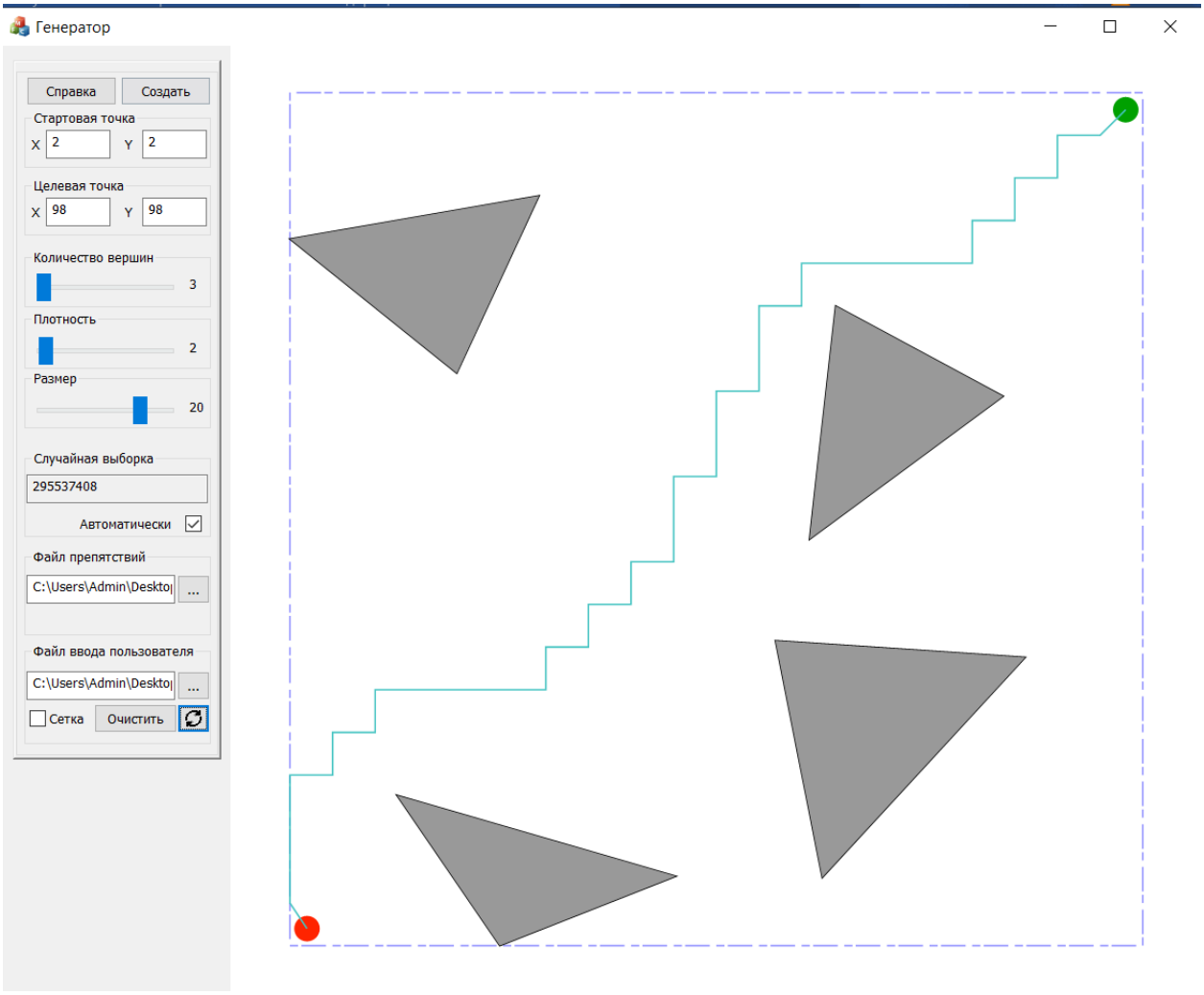

Рисунок 26 – результат теста 2.

Таблица 10 – Сравнение результатов

	Шаг дискретизации	Время заполнения поля	Время работы BFS	Площадь
Тест 1	5	1.16 сек	1.41 сек	1225
Тест 2	5	0.11 сек	0.04 сек	1273

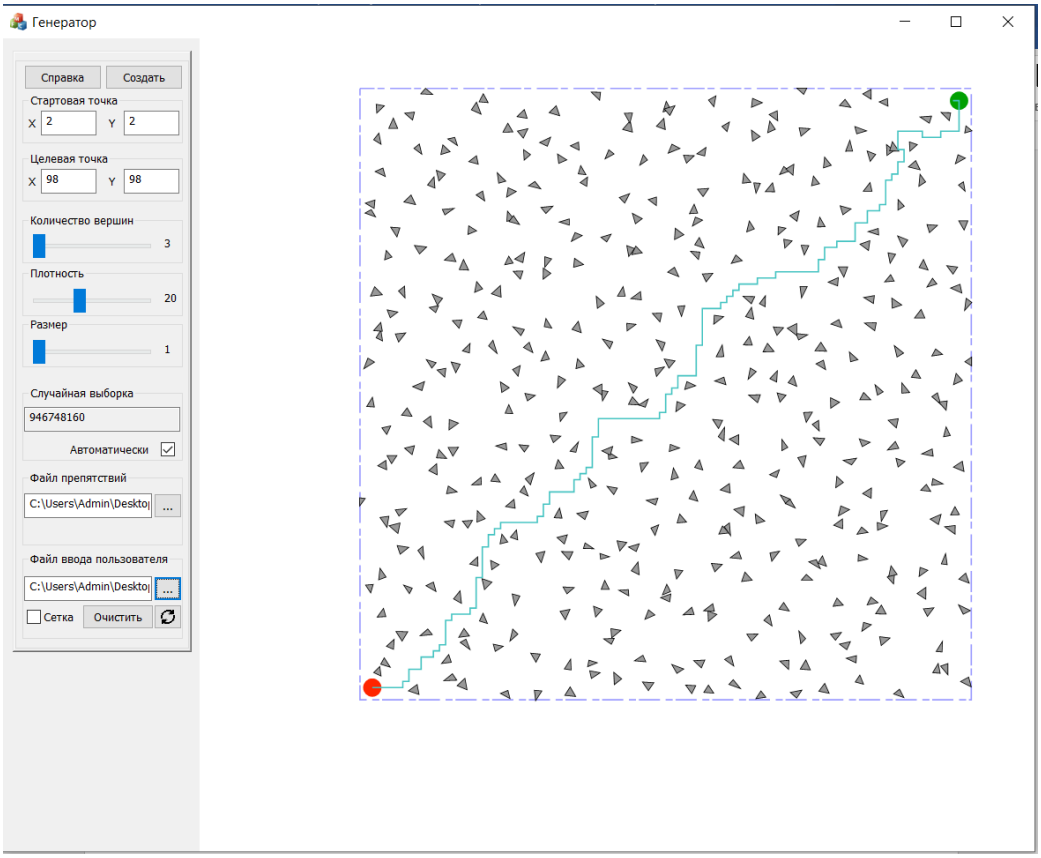

Рисунок 27 – результат работы программы при шаге 1.

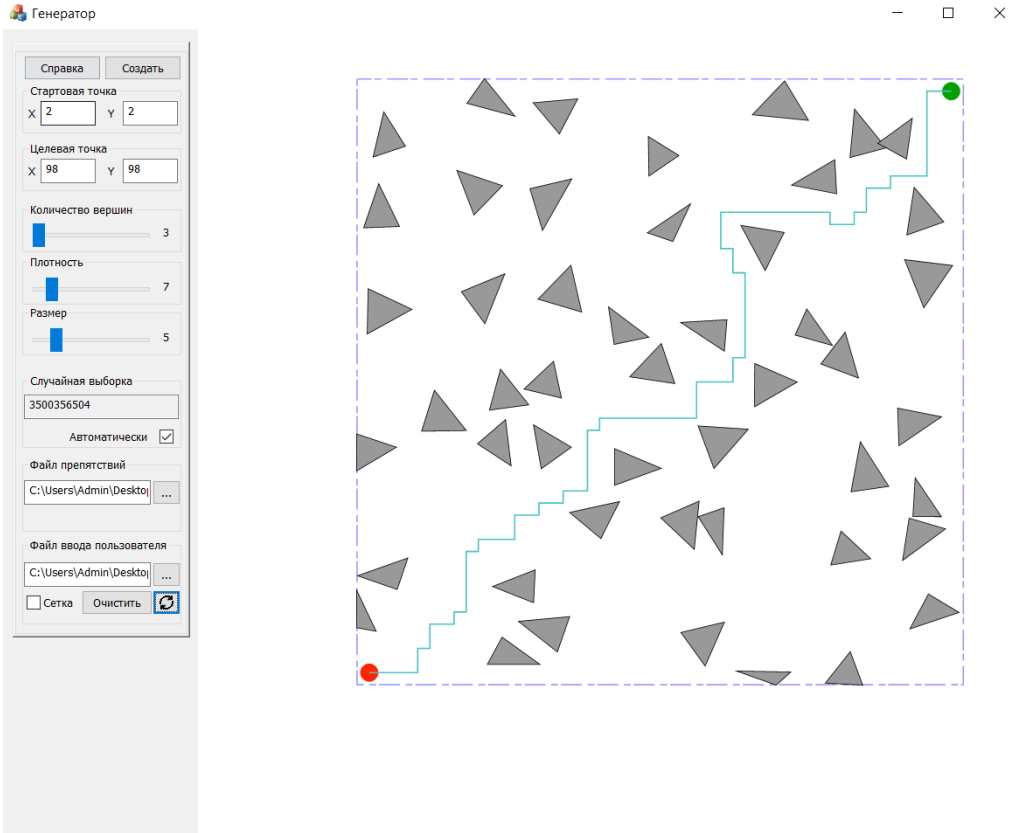

Рисунок 28 – результат работы программы при шаге 2.

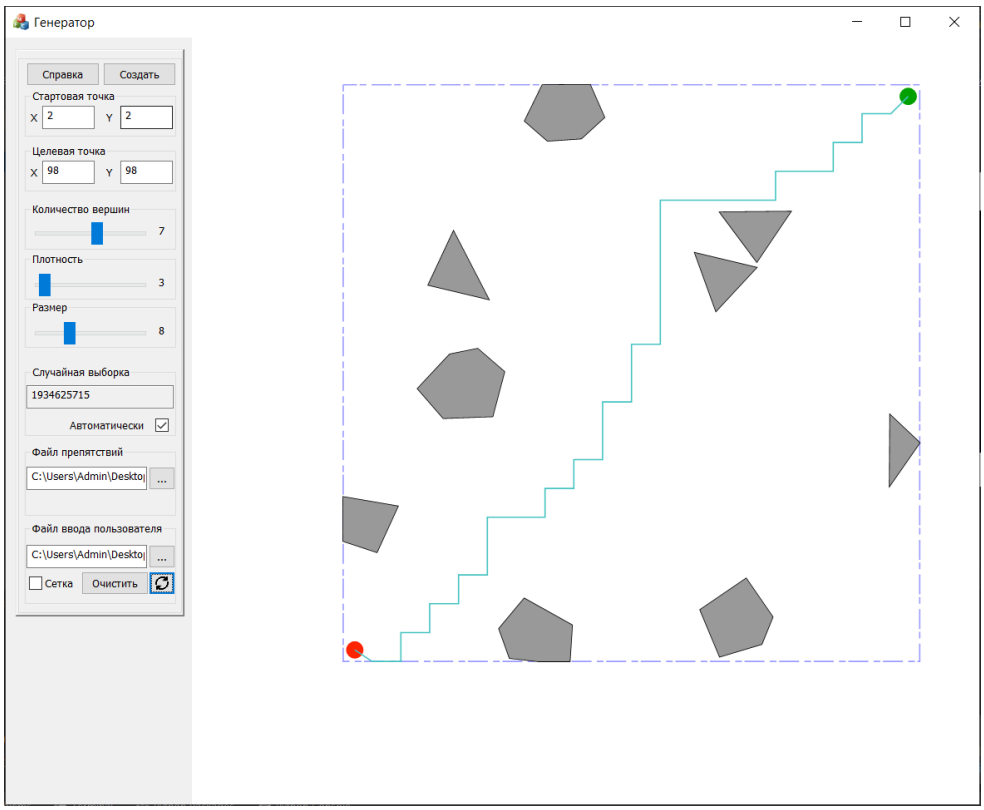

Рисунок 29 – результат работы программы при шаге 5.

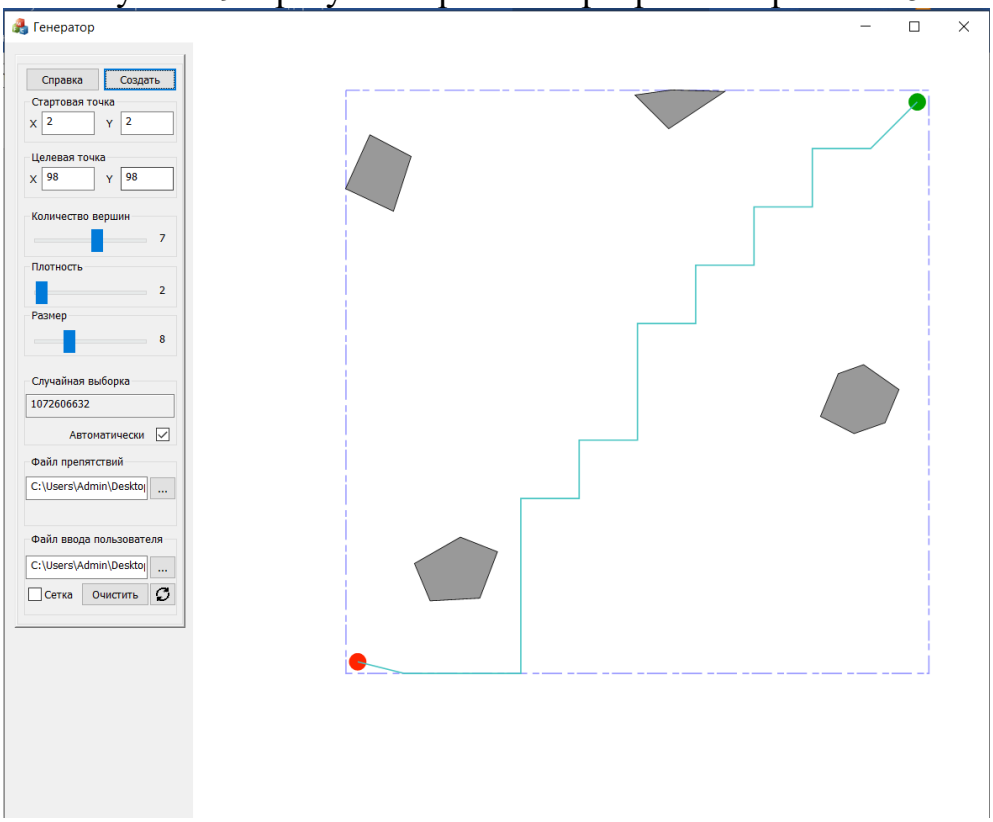

Рисунок 30 – результат работы программы при шаге 10.

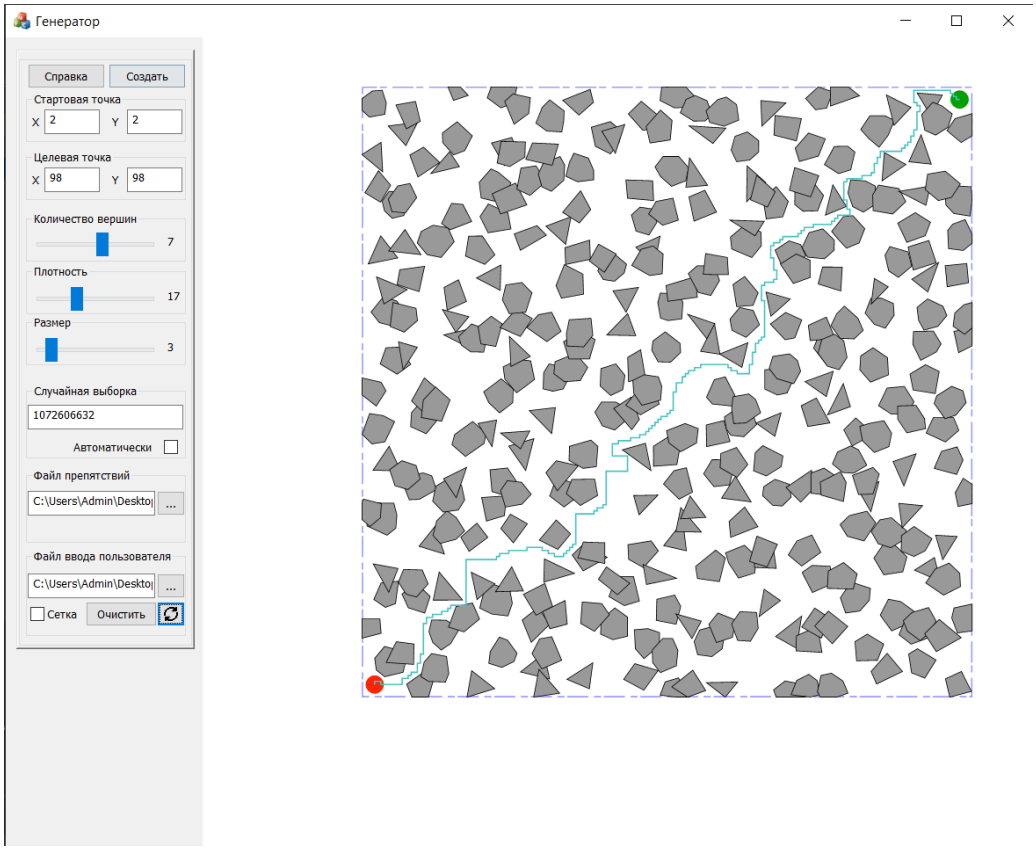

Рисунок 31 – результат работы алгоритма с шагом 0.5.

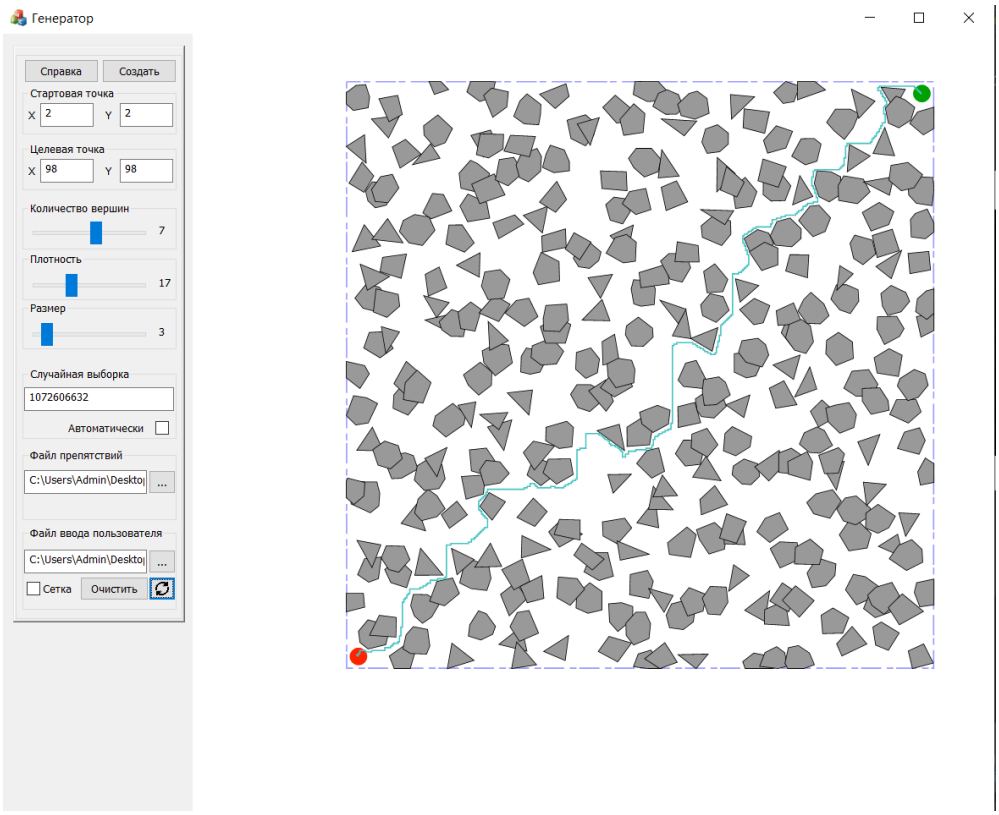

Рисунок 32 – результат работы алгоритма с шагом 0.25.

Таблица 11 – Сравнений результатов тестирования при разном шаге дискретизации

	Шаг дискретизации	Время заполнения поля	Время работы BFS	Площадь
Тест 1	1	157.02 сек	0.67 сек	2500
Тест 2	2	4.85 сек	0.3 сек	1359
Тест 3	5	1.04 сек	0.13 сек	792
Тест 4	10	0.4 сек	0.15 сек	385
Тест 5	0.5	376.82 сек	12.65 сек	4726
Тест 6	0.25	1694.02 сек	56.17 сек	4726

Проведём проверку аварийной остановки алгоритма

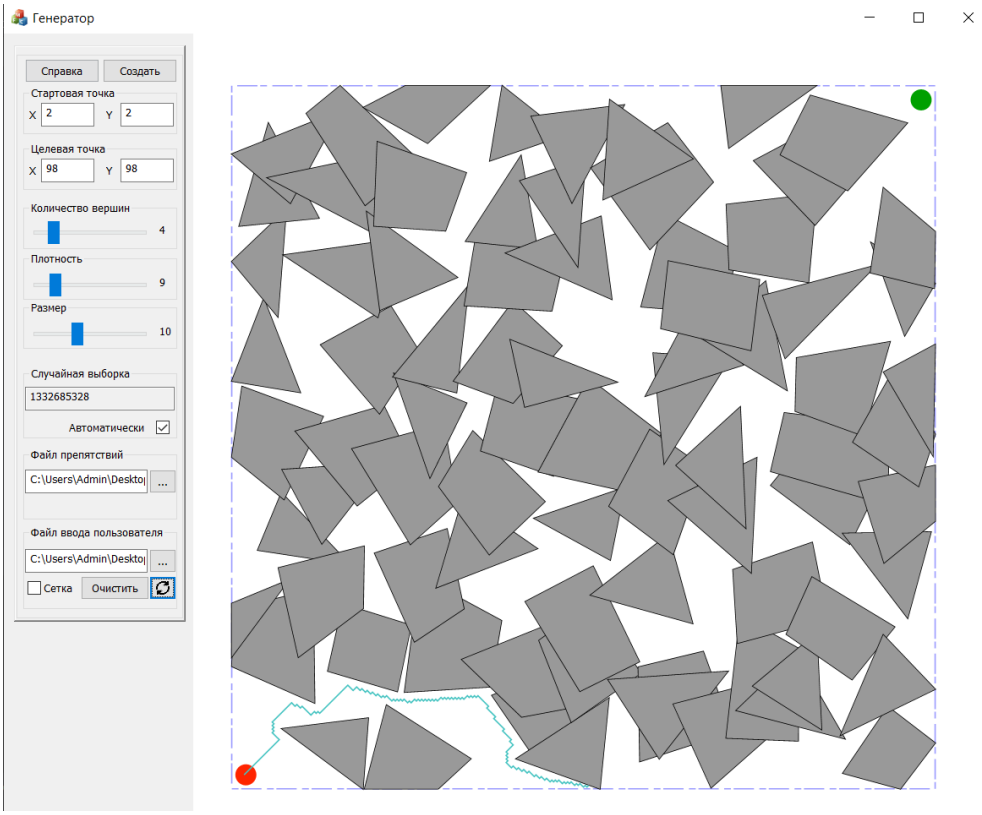

Рисунок 32 – Результат аварийной остановки алгоритма.

Для случаев, когда проход через препятствия невозможен, предусмотрено ограничение по количеству итерации в дереве поиска.

Ограничение по количеству итерации подобрано опытном путём и предполагает собой решений временной эффективности работы алгоритма в случае невозможности прохождения пути.

ЗАКЛЮЧЕНИЕ

В данной выпускной квалификационной работе была рассмотрена задача планирования перемещений для мобильного робота с препятствиями. Были рассмотрены различные методы решения задачи. Было составлено описание и решение алгоритма Метода Искусственных Потенциальных Полей.

Для решения задач была рассмотрена эвристика шага дискретизации. В результате реализации алгоритма на языке Python удалось оптимизировать работу алгоритма в различных средах препятствий, была изучена зависимость соотношений потенциальных уравнений притяжения и отталкивания.

Анализ результатов показал зависимость шага дискретизации от плотности и размера препятствий. Было выявлено, что уменьшение шага дискретизации способствует ускорению процесса решения.

В целом данная работа позволяет сделать вывод о том, что МИПП является эффективным алгоритмом поиска пути. Данная работа имеет практическую значимость для решения задачи планирования пути, а ее результаты могут быть использованы в реальных условиях, связанных с автономным движением роботов и других устройств.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Pang C. Chen, Yong K. Hwang “SANDROS: A Motion Planner with Perfomance Proportional to Task Difficulty” // IEEE International Conference on Robotics and Automation, Nice, 1992, стр. 2346 — 2348
2. Казаков К.А., Семёнов В.А. “Обзор современных методов планирования движения” // Труды ИСП РАН, том 28, вып. 4 , 2016, стр. 241, 244 – 250
3. Лю В. “Методы планирования пути в среде с препятствиями (обзор) // НП “Неикон”, 2018, стр. 15 – 16, 19 – 21, 23, 28 – 31
4. Платонов А.К., Кирильченко А.А., Колганов М.А. “Метод потенциалов в задаче выбора пути: история и перспективы” // ИПМ им. М.В.Келдыша РАН, Москва, 2001, стр. 3, 8, 16 – 19
5. Поиск минимума функций // http://solidstate.karelia.ru URL – http://solidstate.karelia.ru/p/tutorial/meth_calc/files/12.shtml
6. Сороковиков Павел Сергеевич “Разработка и исследование нелокальных алгоритмов параметрической идентификации динамических систем” // ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ УЧРЕЖДЕНИЕ НАУКИ ИНСТИТУТ ДИНАМИКИ СИСТЕМ И ТЕОРИИ УПРАВЛЕНИЯ ИМЕНИ В.М. МАТРОСОВА СИБИРСКОГО ОТДЕЛЕНИЯ РОССИЙСКОЙ АКАДЕМИИ НАУК, Иркутск, 2022, стр. 14, 15
7. А.Н. Божко “Страницы из Motion Planning. Метод искусственных потенциальных полей” // стр. 66 – 75
8. Планирование движения // http://plmpedia.ru/wiki URL – http://plmpedia.ru/wiki/%D0%9F%D0%BB%D0%B0%D0%BD%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5_%D0%B4%D0%B2%D0%B8%D0%B6%D0%B5%D0%BD%D0%B8%D1%8F#:~:text=%D0%97%D0%B0%D0%B4%D0%B0%D1%87%D0%B0%20%D0%BF%D0%BB%D0%B0%D0%BD%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F%20%D0%B4%D0%B2%D0%B8%D0%B6%D0%B5%D0%BD%D0%B8%D1%8F%20%D0%B7%D0%B0%D0%B4%D0

%B0%D0%B5%D1%82%D1%81%D1%8F%20%D0%BE%D0%BF%D0%B8%D1%81%D0%B0%D0%BD%D0%B8%D0%B5%D0%BC,%D0%BD%D0%B0%D1%87%D0%B0%D0%BB%D1%8C%D0%BD%D0%BE%D0%B9%20%D0%B8%20%D1%86%D0%B5%D0%BB%D0%B5%D0%B2%D0%BE%D0%B9%20%D0%BA%D0%BE%D0%BD%D1%84%D0%B8%D0%B3%D1%83%D1%80%D0%B0%D1%86%D0%B8%D0%B9%20%D1%83%D1%81%D1%82%D1%80%D0%BE%D0%B9%D1%81%D1%82%D0%B2%D0%B0.

9. С.М. Тарг “Конфигурационное пространство” // Большая Советская Энциклопедия (БСЭ), 1969 – 1986
10. Задача планирования движения // https://neerc.ifmo.ru URL – https://neerc.ifmo.ru/wiki/index.php?title=%D0%97%D0%B0%D0%B4%D0%B0%D1%87%D0%B0_%D0%BF%D0%BB%D0%B0%D0%BD%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F_%D0%B4%D0%B2%D0%B8%D0%B6%D0%B5%D0%BD%D0%B8%D1%8F#.D0.9E.D0.BF.D1.82.D0.B8.D0.BC.D0.B8.D0.B7.D0.B0.D1.86.D0.B8.D0.BE.D0.BD.D0.BD.D1.8B.D0.B5_.D0.B0.D0.BB.D0.B3.D0.BE.D1.80.D0.B8.D1.82.D0.BC.D1.8B
11. С.М. Караф “ВЕРОЯТНОСТНЫЙ МЕТОД ПЛАНИРОВАНИЯ ТРАЕКТОРИИ ДЛЯ АВТОНОМНОЙ ПЛАТФОРМЫ” // Политехнический молодежный журнал, Москва, 2021, № 09(62), стр. 1, 2
12. А.Б. Филимонов “Новый подход к использованию метода потенциальных полей в мобильной робототехнике” // МИРЭА - Российский технологический университет; Московский авиационный институт (НИУ), Москва, стр. 314-315
13. Расстояние городских кварталов // https://ru.wikipedia.org URL – https://ru.wikipedia.org/wiki/%D0%A0%D0%B0%D1%81%D1%81%D1%82%D0%BE%D1%8F%D0%BD%D0%B8%D0%B5_%D0%B3%D0%BE%D1%80%D0%BE%D0%B4%D1%81%D0%BA%D0%B8%D1%85_%D0%BA%D0%B2%D0%B0%D1%80%D1%82%D0%B0%D0%BB%D0%BE%D0%B2

14. H. Chang and T. Y. Li. “Assembly maintainability study with motion planning.” // In IEEE Int. Conf. Robot. Autom., стр. 1012 – 1019, 1995.
15. Steven M. LaValle “Rapidly-Exploring Random Trees and Motion Strategy Algorithms” // Steven M. LaValle – 1999.

Приложение А

Инициализация объекта поля

```
def __init__(self, iters=2000, start=(1, 1), end=(99, 99)):


    self.size = 100
    self._get_size()

    self.start = self._prepare_start_end(start)
    self.end = self._prepare_start_end(end)

    self.iters = iters
    self.step = 100.0 / self.size
    self.field = np.zeros((self.size, self.size), dtype=Cell)
    self._fill_field()
    self.beta = 1

    self.start_ = start
    self.end_ = end
```

Заполнение препятствиями поле потенциалов

```
for i in range(points_list.shape[0]):
    j = i + 1 - (i // (points_list.shape[0] - 1)) * points_list.shape[0]

    y1, y2 = points_list[i][1], points_list[j][1]
    x1, x2 = points_list[i][0], points_list[j][0]

    # Прохождение по каждой точке на прямой с шагом
    for y in np.arange(self._stepping(min((y1, y2))) + self.step,
self._stepping(max((y1, y2))), self.step):
        x = (y - y1) * (x2 - x1) / (y2 - y1) + x1
        x = self._stepping(x)

        list_of_ones[str(i_pol)].append((y, x))

for y in np.arange(self._stepping(np.min(points_list[:, 1])),
                   self._stepping(np.max(points_list[:, 1])) + self.step,
self.step):
    counter = 0
    cur_line = []
    for yx in list_of_ones[str(i_pol)]:
        if yx[0] == y:
            counter += 1
            cur_line.append(yx[1])
    if counter >= 2:
        for x in np.arange(min(cur_line), max(cur_line), self.step):
            if (y, x) not in list_of_ones[str(i_pol)]:
                list_of_ones[str(i_pol)].append((y, x))
```

Нахождение потенциала отталкивания

```
for distance in cell.distances:
    # cell.capability += self.beta * abs((1 / distance))
    # cell.capability += abs((1 / distance))
    if distance != 0:
        cell.capability += pow((1 / distance), 2)
```

Нахождение потенциала притяжения

```
cell.capability += cell.evcl_distance
```