ВВЕДЕНИЕ

Быстрое развитие техники за последние полвека, привело к бурному развитию робототехники и успешному внедрению промышленных роботов в процесс автоматизированного производства различной продукции. В настоящее время значительно расширилась область применения роботов. Основной проблемой роботов того поколения являлась слабое пространственное описание, благодаря которому робот способен передвигаться и ориентироваться. Развитию в этой области поспособствовало совершенствование навигационных систем и применение их в робототехнике.

Мобильные роботы в настоящее время широко используются в военных, космических, спасательных целях, зачастую, в тех случаях, где человек не сможет проявить все свои качестве в полной мере. Помимо этого возникла еще одна область применения мобильных роботов. Во многих странах начали проводить соревнования, чемпионаты различного уровня между мобильными роботами, созданными студенческими командами ведущих университетов мира. Появился даже новый термин — интеллектуальный спорт, участники которого соревнуются в конструировании наиболее совершенных робототехнических систем.

Цель работы: создание пользовательского интерфейса программы в среде Mathlab для создания дискретного поля, в котором будет происходить планирование траектории с помощью волнового алгоритма. Для этого необходимо изучить принцип работы волнового алгоритма и ознакомиться с теоретическими сведениями о методах траектории мобильного робота.

1 Теоретические сведения о методах планирования траектории мобильного робота

Планирование пути — важнейшая задача в области навигации мобильных роботов. Эта задача включает в основном три аспекта. Во-вторых, этот путь должен обеспечивать движение робота с обходом возможных препятствий. В-третьих, путь должен среди всех возможных путей, удовлетворяющих первым двум требованиям, быть в определенном смысле оптимальным. Для реализации этих задач используют основные методы планирования траекторий мобильного робота. В основном эти методы определяются тем, в какой среде они используются (динамической или статической), или по полноте информации об окружающей среде.

Наиболее распространенным в последнее время являются м**етоды на основе графов.** Основные методы этого класса показаны отражают состояния, в которых может находиться робот: каждый узел представляет одно состояние робота. Состоянием может быть положение, угол ориентации, скорость или ускорение робота. Переходы между состояниями характеризуются функцией затрат. Это позволяет выделить путь, который имеет минимальную общую стоимость достижения целевого состояния.

В **методах на основе клеточной декомпозиции** основной идеей является дискретизация окружающей среды. В различных реализациях этой идеи можно выделить две группы: приближенная и точная клеточные декомпозиции. Приближенная клеточная декомпозиция реализуется с помощью сетки покрывающей пространство. Сетка проста в использовании, и её легко создать и поддерживать. Однако метод сеток имеет серьезный недостаток: увеличение трудоемкости при уменьшении шагов сетки. Такое увеличение особенно заметно в окружающей среде большого объема.

Один из широко распространенных подходов к планированию траекторий — использование потенциальных векторных полей, так называемые **методы потенциальных полей.** Общая идея методов состоит в движении вдоль векторных линий векторного поля, потенциальная функция которого отражает конфигурацию препятствий и их форму, а также цель движения. Указанный подход подходит и в двумерном, и в трехмерном случае. Один из широко распространенных подходов к планированию траекторий — использование потенциальных векторных полей. Общая идея методов состоит в движении вдоль векторных линий векторного поля, потенциальная функция которого отражает конфигурацию препятствий и их форму, а также цель движения. Указанный подход подходит и в двумерном, и в трехмерном случае.

С помощью **оптимизационных методов** планирование пути рассматривается как оптимизационная задача. Препятствия будут описываться некоторыми ограничениями, а качество допустимой траектории должно оцениваться некоторым функционалом. В результате возникает задача оптимального управления, которая не только обеспечивает траекторию объекта в обход препятствий, но и позволяет выбрать в некотором смысле лучший вариант, например, по скорости прохождения. Подход, основанный на оптимальном управлении, эффективен для простых, в частности, линейных систем, а для сложных нелинейных систем его реализовать труднее.

Методы интеллектуальных алгоритмов, таких как Муравьиный алгоритм, Искусственная нейронная сеть, Метод роя частиц и Реактивные методы находят широкое применение для планирования траектории мобильного робота. Задача планирования пути является одной подзадач автоматического управления роботом. Робот должен иметь способность решать задачу планирования пути в реальных условиях окружающей среды без вмешательства человека. Естественно в этой ситуации использовать алгоритмы поведения, встречающиеся в живой природе. Эти алгоритмы имитируют поведение или мышление человека, а также некоторых биологических сообществ.

Такое большое количество методов может натолкнуть на непонимание, какой же из них наиболее эффективен, однако эта проблема не должна рассматриваться обособлено, должны учитываться факты картографии, местности, навигационного оборудования и скорости вычисления алгоритма. Современный же алгоритм планирования пути должен в себе интегрировать преимущества различных существующих методов и быть свободен от недостатков отдельных методов.

2 Выбор метода поиска траектории пути мобильного робота и его математическое описание

Ознакомившись с оновными методами планирования траектории мобильного робота, составим алгоритм на волновой основе, который будет находить кратчайшее расстояние между точками. Для этого применим методы, которые используются на основе графов и клеточной дискретизации.

Волновой алгоритм или Алгоритм Ли является одним из видов алгоритмов обхода графов в ширину. Прежде чем перейти к принципу работы алгоритма, необходимо ознакомиться с математическим определением графа и каким образом он задается для работы алгоритма.

Граф G – пара множеств, G=(V,E), где V – произвольное непустое множество (множество вершин), а E – множество пар различных вершин (множество ребер).

$V = \{v1, \ldots, v_n\}$ – конечное множество вершин графа G и $E = \{(vi, vj) | vi, vj \in V$ и $vi\ 6 = vj\}$ - конечное множество ребер графа G, тогда G называется (n, m)-графом.

$|V| = n$ – число вершин графа G, $|E| = m$ - число ребер графа G.

Если $(vi, vj) = (vj, vi)$ при $i \neq j,\ i,\ j = 1, 2, \ldots, n$ то граф G называется неориентированным.

Два ребра e1 и e2 в графе G называются смежными, если они инцидентны одной и той же вершине $v \in V$, то есть если $e1 = (v, v1) \in E$ и $e2 = (v, v2) \in E$.

Если $(vi, vj) \neq (vj, vi)$при $i \neq j, i, j = 1, 2, \ldots, n,$ то граф G называется ориентированным (орграф).

Графический способ задания графа. Изображение графа *G=(V,E)* получают следующим образом: каждой вершине $vi \in V$ соответствует точка на плоскости, причем если $(vi, vj) \in E$, то точки, соответствующие вершинам vi и vj, соединяются линией.

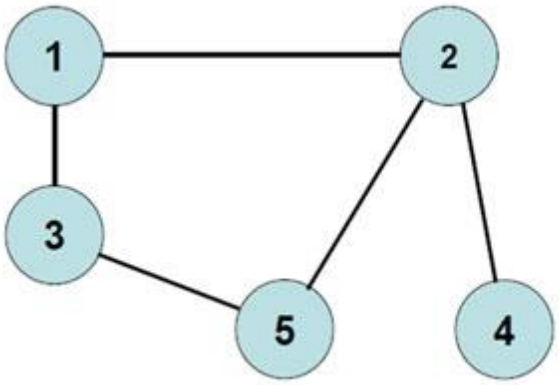

Рисунок 1 – Пример геометрического изображения графа

Помимо того, что граф является множеством вершин и ребер, которые между собой инцидентные и смежные, эти множества необходимо правильно представить, чтобы компилятор смог «распознать» эти объекты. Для этого могут использоваться представление в виде матрицы смежности или представление в виде списка смежности. В данном алгоритме будет использован список смежности.

Представление графа *G=(V,E)* в виде списка смежности использует массив из |V| списков для каждой вершины из множества V. Для каждой вершины $v \in V$ список смежности содержит все вершины v такие, что $(vi$ и $vj) \in E$, массив состоит из всех вершин, смежный с v в графе G. Поскольку списки смежности представляют собой ребра графа, то массив, содержащий вершины, рассматривается как атрибут графа, так же, как мы рассматриваем множество ребер E. Для неориентированного графа сумма длин всех списков смежности равна $2|E|$, поскольку ребро $(vi$ и $vj) \in E$, будучи неориентированным, появляется как в списке смежности vi, так и в списке смежности vj.

Потенциальный недостаток представления с помощью списков смежности заключается в том, что нет более быстрого способа определить, является ли данное ребро (vi, vj) в графе, чем поиск vj в массиве вершин.

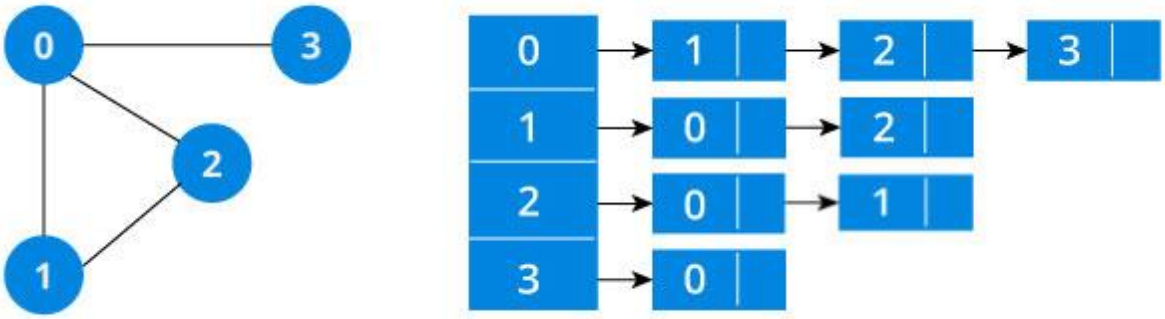

Рисунок 2 – Пример представления графа списком смежности

Теперь, исходя из определения графа и возможности представления его как списка смежности, перейдем к реализации волнового алгоритма.

В основном волновой алгоритм используется при компьютерной трассировке (разводке) печатных плат, соединительных проводников на поверхности микросхем. Другое применение волнового алгоритма — поиск кратчайшего расстояния на карте в компьютерных стратегических играх.

Алгоритм работает на дискретном рабочем поле (ДРП), представляющем собой ограниченную замкнутую линию фигур (в данном случае – прямоугольных), которая разбита на прямоугольные ячейки (в данном случае – квадратные). Перемещение из одной ячейки в другую является единичным перемещением, определенным по горизонтали и вертикали. Множество ячеек ДРП разбивается на подмножества: те, что можно пройти (свободные), те, что пройти нельзя (препятствия) и те, которые уже были пройдены (ячейки предков). В ДРП определяется стартовая ячейка, то есть откуда начнется работа алгоритма, и финишная ячейка – где алгоритм прекращает поиск.

Дискретное рабочее поле можно представить в виде графа, где вершины каждой ячейки – это вершины графа, а стороны каждой ячейки – это его ребра. Поскольку ДРП в нашем случае нет никаких направлений ребер, то этот граф будет неориентированным. Для того, чтобы составить список смежности для этого графа, необходимо пронумеровать вершины.

Этот граф довольно легко представляется в виде списка смежности, причем у каждой вершины есть как минимум две смежные вершины, а максимум их может быть 4. Поскольку мы считаем, что в нашем дискретном поле находятся помимо свободных вершин ещё и вершины-препятствия, вершины-предки, а также исходная вершина и конечная вершина, которые уже были

пройдены, необходим маркер, который сможет идентифицировать каждую вершину.

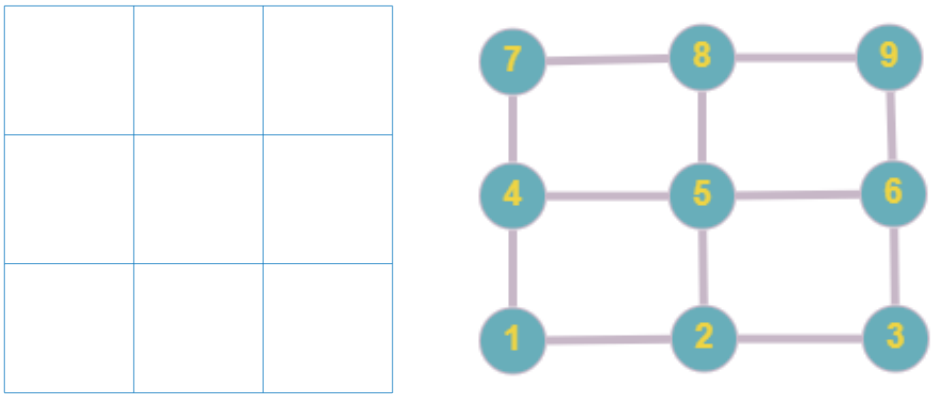

Рисунок 3 – Пример дискретного рабочего поля и представление его в виде графа

Для этого мы применяем к вершине графа $v \in V$ атрибут $v.obstacle$, который может быть либо 0 для свободной вершины, либо -1 - для вершины-препятствия, либо 1- для стартовой вершины. Смежность каждой отдельной вершины будем хранить в атрибуте $v.sm$, который будет являться массивом. Для удобства поместим наше дискретное рабочее поле в декартову систему координат, где каждой вершины будет соответствовать пара координат (x, y), которые будут занесены в атрибут $v.coordinate$. Поскольку необходимо определить кратчайшее расстояние между двумя вершинами, а расстояние между двумя соседними вершинами принимается за 1, добавим атрибут $v.d$, обозначающий количество ребер, отделяющих две вершины. Чтобы составить иерархию поиска вершин, добавим атрибут $v.parent$, который будет указывать, от какой вершины исходит данная вершина.

Перейдем к описанию работы волнового алгоритма. Для начала задаются стартовая и финишная вершина, атрибут стартовой позиции $v.obstacle = 1$. С этой вершины начинается поиск в ширину, при котором происходит обход всех ребер графа G для «открытия» вершин всех вершин, достижимы из стартовой вершины, вычисляя при этом расстояние (минимальное количество ребер) от стартовой вершины до каждой достижимой из нее вершины. Значение расстояния заносится в атрибут вершины $v.d$. Алгоритм не перейдет на расстояние d+1, пока не найдет все вершины, находящиеся на расстоянии d от стартовой вершины. Соответственно, для всех вершин на расстоянии d=1,

предком будет являться стартовая вершина, поскольку с нее и начинается поиск. После того, как вершина была определена, в ее атрибут $v.obstacle$ заносится 1, чтобы при следующем попадании на вершину, алгоритм ее пропускал. Препятствие отличается от других вершин лишь атрибутом $v.obstacle = -1$. Благодаря этому атрибуту алгоритм не будут искать смежные вершины для препятствия, то есть «обрубит» любое ребро из смежной вершины, перекрыв путь. Алгоритм действует до тех пор, пока не будет найдена финишная вершина, то есть пока ее атрибут $v.obstacle$ не станет равным 1 или пока не будут определены все вершины, в таком случае пути нет, что говорит о перекрытии финишной вершины.

Для работы с множеством вершин, которые еще не определены, будет использоваться такая структура данных как очередь. Так как в Matlab отсутствует очередь, используется пользовательская функция, выполняющая все свойства очереди. Важность очереди заключается в том, что соблюдается порядок исследования вершин смежности. К примеру, если у первой вершины на расстоянии d имеется две смежные вершины, то перейдя на расстояние d+1, алгоритм начнет исследовать сначала смежные вершины этих двух вершин.

Определив расстояние каждой свободной вершины от стартовой вершины, начинается поиск кратчайшего расстояние. Для этого применяется атрибут $v.parent$. Сначала определяется значение атрибута $v.parent$ для финишной вершины, а для стартовой вершины изначально $v.parent = 0$. Понижая $v.parent$ на единицу, мы определим предшествующую вершину, смежную с финишной. Этот поиск продолжается до тех пор, значение атрибута $v.parent$ не станет равным 0, что обозначит приход на стартовую позицию. Для отображения траектории на дискретном поле будут использоваться координаты вершин, которые однозначно определяются благодаря атрибуту таким образом.

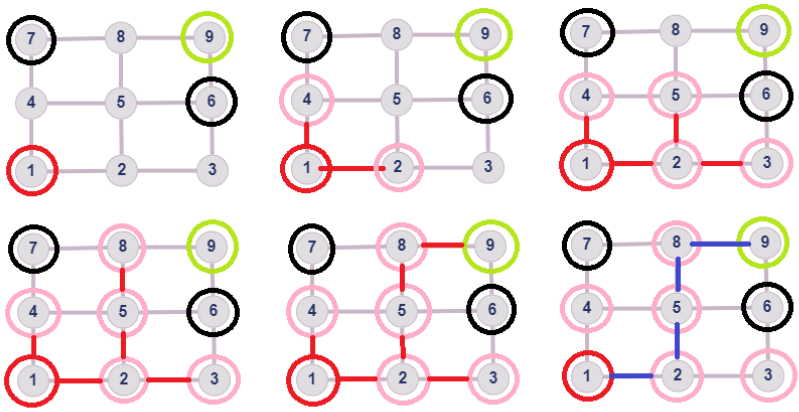

Рисунок 4 – Пример выполнения волнового алгоритма на графе:

Программа позволяет на выбор или самому задать препятствия, или построить их случайным образом, нажимая на график левой кнопкой мыши поля на нужную ячейку. Можно выбрать подходящую размерность поля, определить начальные и конечные вершины, через которые будет искаться кратчайший путь.

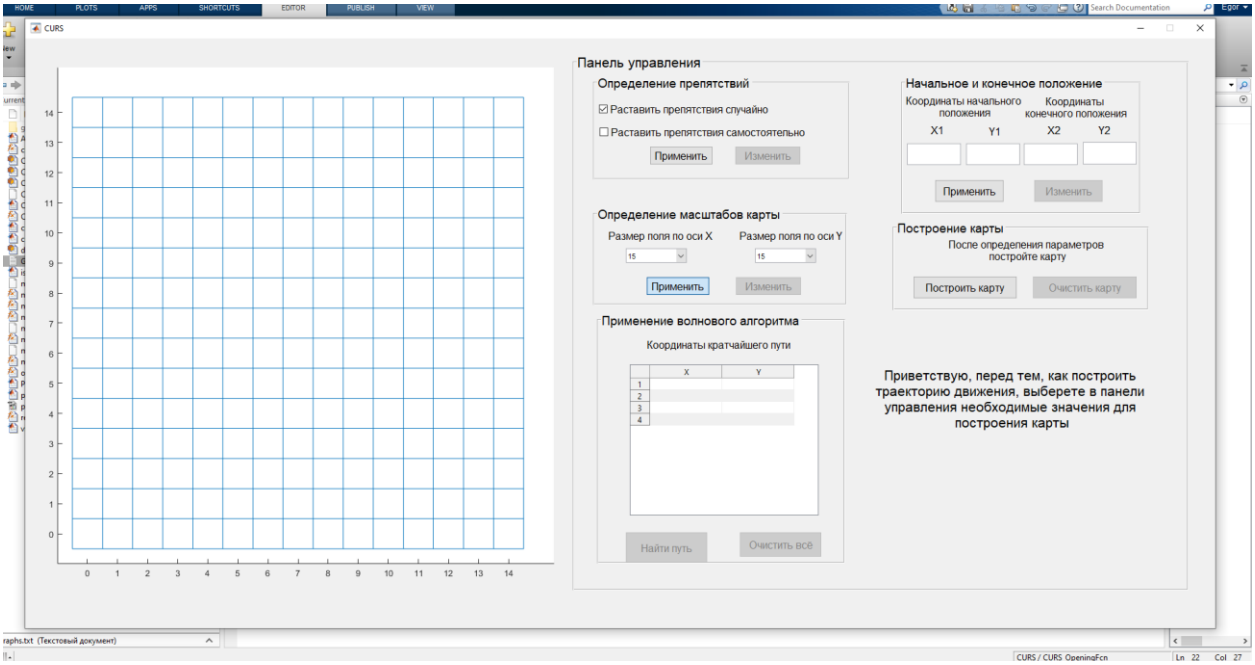

Рисунок 5 – Рабочий интерфейс при запуске

При нажатии кнопки «Построить карту» создается необходимое рабочее поле, на котором можно искать путь. При нажатии кнопки «Найти путь» начинает работать волновой алгоритм с интерфейсным представлением на рабочем поле.

Программа содержит текстовый помощник, который помогает разобраться в панели управления. При нажатии клавиш текст меняется, указывая на дальнейшие действия.

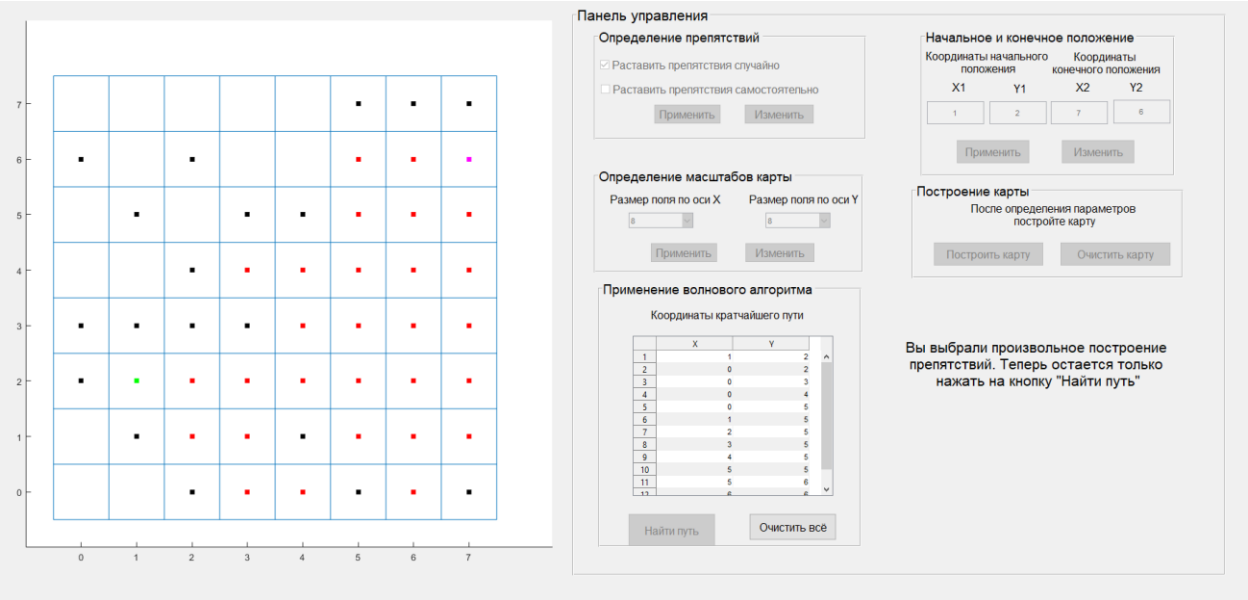

Рисунок 6 – Распространение волны

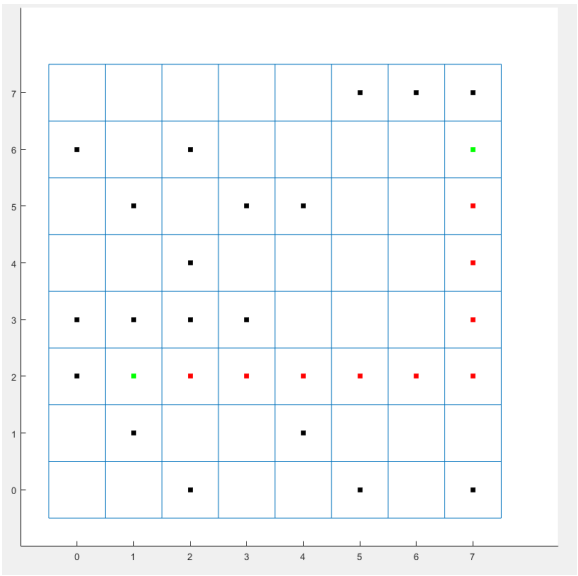

Рисунок 7 – Построение кратчайшего пути между точками

ЗАКЛЮЧЕНИЕ

Среда программирования Matlab прекрасно подходит для реализации таких задач, как определение траектории, местоположения и описание карты местности благодаря обширному множеству функций и операторов. В дальнейшем хотелось бы углубиться в эту тему, ознакомившись с другими алгоритмами, применить эти алгоритмы на практике, связать их с навигационными системами для определения траектории в реальных условиях.

При выполнении данной работы мне потребовались все знания, которые были получены на протяжении всего курса «Алгоритмизации и обработки данных»: это работа с файлами, структурами, массивами, интерфейсными объектами, а также это создание функций и их применение.

В результате данной курсовой работой я ознакомился с одним из базовых алгоритмов в планировании траектории мобильного робота, включающий в себя работу с графами и структурами данных. Алгоритм позволяет определить кратчайшее расстояние в дискретном рабочем поле для любой прямоугольной области. Для его реализации были использованы 2 метода: метод графов и клеточной дискретизации. Описанный математически граф заключал в себе данные для работы алгоритма, а дискретизации позволила упростить поле, разделив его на множество дискретных ячеек. Преимущества этого алгоритма в том, что он однозначно определяет траекторию в простых полях. К недостаткам можно отнести большое время выполнения и неоптимизацию маршрута: у алгоритма нет никакой возможности сократить поиски, поскольку он всегда будет перемещаться волной, у которой нет распределения в сторону препятствия.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

[1] Кармен, Т. Алгоритмы: построение и анализ Пер. с англ / Кармен Т. –М: – ООО “И.Д,Вильямс”, 2013. – 527 с., с ил. – Парал. тит. Англ

[2] Лю В. Методы планирования пути в среде с препятствиями (обзор) - Электронный ресурс // – Режим доступа: https://www.mathmelpub.ru/jour/article/view/98/105

[3] CyberForum, электронный форум [Электронный ресурс] // – Режим доступа: https://www.cyberforum.ru/

[4] Хабр, электронный форум [Электронный ресурс] // – Режим доступа: https://habr.com/ru/all/

[5] Куликова, В. Н. Лабораторные работы по программированию в Matlab / В. Н. Куликова. – 2021г.

[6] Документация MATLAB [Электронный ресурс] // ЦИТМ Экспонента. – Режим доступа: https://docs.exponenta.ru/R2019a/documentation-center.html