УДК 656.078 DOI: 10.31799/2077-5687-2023-3-220-225

РЕШЕНИЕ ВОПРОСА МАРШРУТИЗАЦИИ БЕСПИЛОТНОГО ВОЗДУШНОГО СУДНА С ПОМОЩЬЮ АЛГОРИТМА ДЕЙКСТРЫ

Е. С. Виноградова
Санкт-Петербургский государственный университет аэрокосмического приборостроения

В статье описывается значимость развития беспилотных авиационных систем как части единой транспортной системы и подчеркивается, что одним из решений в данной области может быть разработка интеллектуальных алгоритмов управления полетом БАС. В статье представлено применение алгоритма Дейкстры для планирования траектории БАС на условной городской местности. В результате была разработана программа, реализующая алгоритм Дейкстры для дрона и предоставляющая матрицу кротчайших расстояний между вершинами графа.

Ключевые слова: беспилотная авиационная система, квадрокоптер, маршрутизация, алгоритм Дейкстры.

Для цитирования:
Виноградова, Е. С. Решение вопроса маршрутизации беспилотного воздушного судна с помощью алгоритма Дейкстры / Е. С. Виноградова // Системный анализ и логистика. – 2023. – № 3(37). – с. 220 – 225. DOI: 10.31799/2077-5687-2023-3-220-225.

SOLVING THE ISSUE OF ROUTING AN UNMANNED AIRCRAFT USING DIJKSTRA'S ALGORITHM

E. S. Vinogradova
St. Petersburg State University of Aerospace Instrumentation

The article describes the importance of the development of unmanned aircraft systems as part of a unified transport system and emphasizes that one of the solutions in this area may be the development of intelligent flight control algorithms UAS. The article presents the application of Dijkstra's algorithm for planning the UAS trajectory in a conditional urban area. As a result, a program was developed that implements Dijkstra's algorithm for the drone and provides a matrix of the shortest distances between the vertices of the graph.

Keywords: unmanned aircraft system, quadcopter, routing, Dijkstra algorithm.

For citation:
Vinogradova, E. S. Solving the issue of routing an unmanned aircraft using Dijkstra's algorithm / E. S. Vinogradova // System analysis and logistics. – 2023. – № 3(37). – p. 220 – 225. DOI: 10.31799/2077-5687-2023-3-220-225.

Введение

Во всём мире всё ещё нет полностью гармонизированной транспортной системы. Нет общих для всех правил и стандартов, которые давали бы понимание о роли БВС в воздушном пространстве, их «поведение» относительно других воздушных судов, а если говорить про БВС в условиях городской среды, то и их место в городе, нормы их маршрутов движения и систем распознавания. Эти вопросы были подняты на форуме «РОССИЙСКАЯ СОВРЕМЕННАЯ АВИОНИКА – 2023» от компании АО «Навигатор» на пленарной дискуссии «Интеграция БАС в общее воздушное пространство», в которой приняли участие представители ООО «Геоскан», ПАО «Корпорация «Иркут», АО «РИВР», АНО «Университет Национальной технологической инициативы 2035», ООО «Флай Дрон», АО «КТ-Беспилотные системы» и Ассоциации вертолетной индустрии.

Значимость системности в деятельности по развитию БАС подчеркнул, и советник ПАО «Иркут» Олег Косолапов «Нужно давать рынку не отдельные решения, а комплекс, который будет состоять не только из беспилотников, но и систем их управления, необходимых сервисов для работы с БВС в автоматическом режиме» [1]. Таким образом, создание эффективных методов формирования маршрутов и профилей полёта БВС является актуальной задачей, требующей дополнительных исследований и разработок. Одним из решений может

быть разработка интеллектуальных алгоритмов управления полётом БАС [2], принимающих во внимание ситуацию в полётном пространстве, тактическую обстановку и другие аспекты в зависимости от сферы применения.

Описание подхода

Алгоритм Дейкстры относится к традиционным алгоритмам [3, 4], это значит, что для того, чтобы использовать его в решении вопроса маршрутизации БАС, в частности, квадракоптера, нужно предварительно преобразовать карту местности в граф, который отображает карту местности, где вершины графа – точки на карте, а ребра – возможные маршруты полёта между точками, учитывая стоимость каждого ребра, которая зависит от расстояния между вершинами, скорости квадракоптера и других факторов. После нахождения кратчайшего пути, необходимо задать квадракоптеру последовательность близко расположенных точек, через которые он может пролететь.

Однако, алгоритм Дейкстры может не учитывать различные ограничения, например, наличие препятствий или ограничения высоты полёта, которые могут повлиять на безопасность полёта квадрокоптера. Поэтому необходимо указать все ограничения и требования конкретной задачи и адаптировать алгоритм соответственно для безопасного управления полётом квадракоптера.

Предположим, у нас есть карта городской местности, которая схематично изображена на Рис. 1 с учётом препятствий, которые находятся на возможных маршрутах.

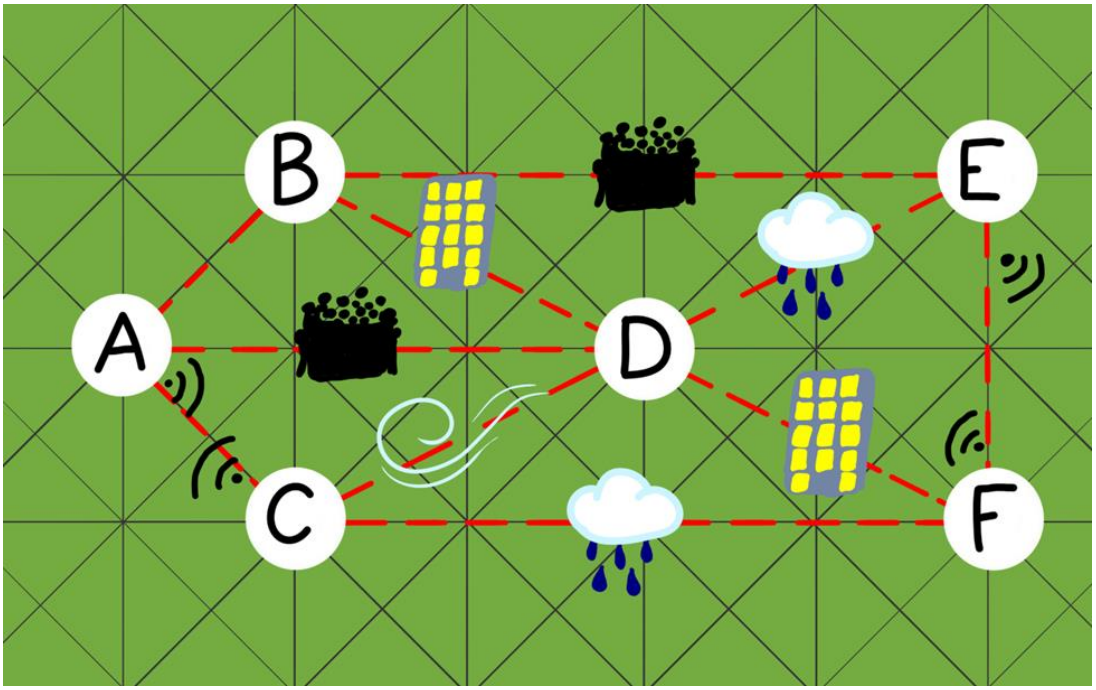

Рис. 1. Схематичная карта городской местности с учётом препятствий для квадракоптера на возможных маршрутах

На Рис. 1 красная пунктирная линия – возможные маршруты полёта между точками; A, B, C, D, E, F – вершины графа и точки, которые должен облететь квадракоптер; на промежутке A-B препятствий нет, на промежутках A-C и E-F находятся городские «глушилки» для дронов, на промежутках A-D и B-E – людные места, B-D и D-F – высотные здания, D-E и C-F – дожди, C-D – ветер скоростью больше 10 м/с.

Каждому из препятствий мы присвоили весовой коэффициент, которые представлены в таблице 1.

Таблица 1 – Весовые коэффициенты препятствий

Препятствие	Весовой коэффициент
Отсутствие препятствия	1
Дождь	2
Ветер скоростью больше 10 м/с	3
Высотное здание	4
Людное место	5
Городские «глушилки»	6

С учётом расставленных коэффициентов преобразуем Рис. 1 в граф, который представлен на Рис. 2.

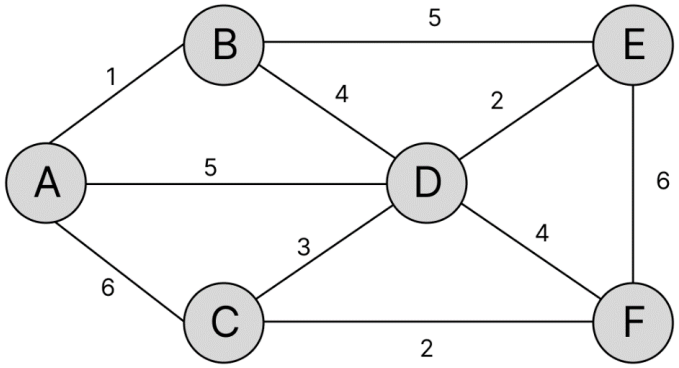

Рис. 2. Граф на основании карты местности

Реализация алгоритма

Алгоритм реализовывался в среде *MATLAB* [5]. Написанная нами функция *distances* предполагает, что в результате мы получим карту расстояний между начальным и конечным вершинами графов. Исходя из этой логики, для начала нужно узнать, сколько всего вершин мы предоставили.

Следом от нас требуется инициализировать расстояние до всех точек как бесконечность, а все узлы отметить как не посещённые; инициализировать расстояние до первой точки равным нулю.

Определимся, при каком условии выполнение алгоритма завершится. Когда одна из *N* вершин будет посещена, ей присваивается значение 1, постепенно вершин со значением 0 не останется, что значит, все *N* вершин станут равны 1. Тогда в качестве условия мы ставим следующее выражение: *while sum(visited) < N*. Алгоритм выполняется до тех пор, пока сумма посещённых вершин не равна общему количеству вершин в графе.

В цикле *while* выполняются следующие действия:

1. Определение не посещённых вершин;
2. Нахождение вершины с наименьшим значением расстояния, она становится текущей точкой, а её расстояние – текущим расстоянием;
3. Учитывая расстояния до текущей вершины, обновить расстояния до всех её соседей, если новый путь к соседней вершине короче, чем прежний путь;
4. Отметить текущую точку как посещённую.

Таким образом, в функцию *distances* вносится матрица map, которая соответствует имеющемуся графу и стартовая точка.

Матрица *map* симметрична, что говорит нам о двунаправленных рёбрах графа. Размеры матрицы зависят от количества вершин, т.е. имеет размер *[N×N]*. Каждая строчка соответствует очерёдности вершины от A до F, от которой смотрят расстояния до остальных вершин, каждый столбец – вершина, расстояние до которой вносится в ячейку.

Исходя из условий, которые были внесены в функцию *distances*, расстояние в исходной вершине обозначается как 0, если от начальной вершины до конечной нет прямого ребра, такое расстояние считается бесконечно большим. Тогда для случая на Рис. 2 матрица *map* выглядит следующим образом.

$$map = \begin{bmatrix} 0 & 1 & 6 & 5 & Inf & Inf \\ 1 & 0 & Inf & 4 & 5 & Inf \\ 6 & Inf & 0 & 3 & Inf & 2 \\ 5 & 4 & 3 & 0 & 2 & 4 \\ Inf & 5 & Inf & 2 & 0 & 6 \\ Inf & Inf & 2 & 4 & 6 & 0 \end{bmatrix};$$

Осталось лишь использовать уже написанную ранее функцию, чтобы получить карту расстояний между начальным и конечным вершинами графов. Удобство функции *distances* в том, что, если нам нужно использовать другие стартовые точки, достаточно поменять значение второго аргумента функции.

Если стартовой вершиной является вершина A, тогда граф, с которым будет работать алгоритм представлен на Рис. 3. На Рис. 3 не указаны бесконечно большие расстояния для улучшения читабельности графа. Результат работы программы представлен следующим образом:

```
>> distances = dijkstra(map,1)
distances =                                                    (1)
    0     1     6     5     6     8
```

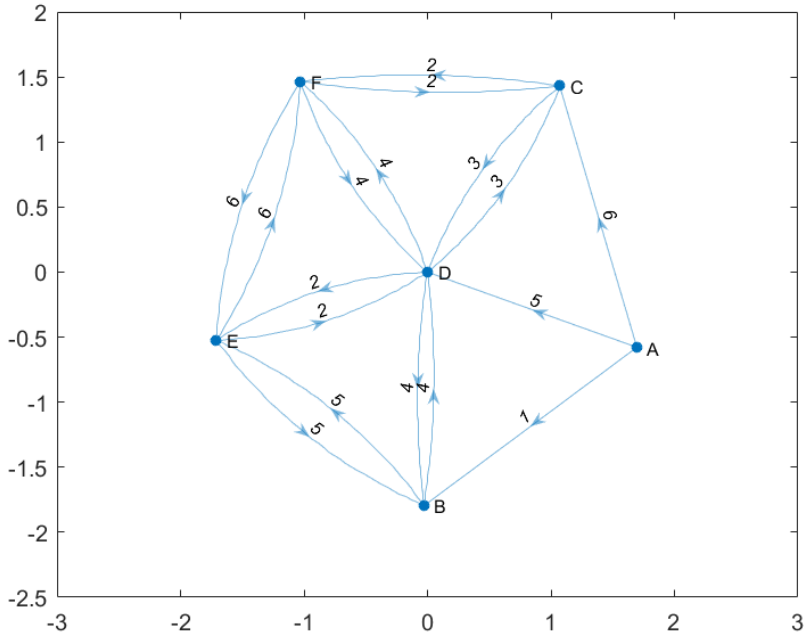

Рис. 3. Граф для реализации разработанного алгоритма в случае, когда стартовой вершиной является вершина A без указания бесконечно длинных расстояний

Аналогично будет выглядеть граф для случая, когда стартовой вершиной является вершина E, а результат работы программы представлен ниже:

```
>> distances = dijkstra(map,5)
distances =                                                    (2)
    6    5    5    2    0    6
```

Проверка корректной работы программы

Выполним проверку работы нашего алгоритма при помощи встроенных функций среды *MATLAB*. Зададим переменной G направленный граф, который будет включать в себя информацию из матрицы map и названия вершин графов.

```
>> G = digraph(map,node_names);
```

Теперь используем функцию *shortestpath*, которая позволяет находить кротчайшие расстояния между двумя конкретными вершинами графа. В выходные значения занесём две переменных: *dist* и *path*. *dist* – кратчайшее расстояние, которое можно сравнить с результатами работы нашей функции, *path* – узлы пути кротчайшего расстояния. Так как выше в качестве примера работы программы использовались случаи, когда A является стартовой вершиной и когда E является стартовой вершиной, тогда и для проверки мы будем искать расстояния между двумя этими функциями. Обратим внимание, что по результатам работы программы (1) и (2) кротчайшее расстояние между A и E равно 6.

```
>> [path,dist] = shortestpath(G,1,5)

path =
    1    2    5

dist =
    6
```

Что и требовалось доказать, переменная dist ровна 6, что соответствует результатам работы, разработанной нами функции distances. Таким образом, можно говорить о корректности работы нашей программы.

Заключение

Целью этой работы было разобрать использование алгоритма Дейкстры в вопросах планирования траектории для БВС. Было проведено моделирование использования алгоритма на условной городской местности. Во внимание были взяты как преимущества, так и недостатки алгоритма. Граф построен на основании карты предполагаемой местности, после оценки препятствий на пути квадракоптера каждое ребро получило свою стоимость, которая влияет на приоритетность выбора маршрута.

В результате была разработана программа, которая реализует алгоритм Дейкстры для БВС и предоставляет матрицу кротчайших расстояний между начальными и конечными вершинами графов. Матрица кротчайших расстояний представляет собой одномерный массив, в котором каждый столбец соответствует определённой вершине. Программа позволяет получить матрицу кротчайших расстояний от любой из вершин графа. Корректность расчёта кротчайших расстояний была доказана с помощью встроенных функций среды MATLAB.

Таким образом, можно говорить, что при в неменяющихся условиях окружения, которые предварительно были оценены и преобразованы в граф, с помощью разработанного алгоритма беспилотное воздушное судно способно решить задачу маршрутизации.

СПИСОК ЛИТЕРАТУРЫ

1. Форум «РОССИЙСКАЯ СОВРЕМЕННАЯ АВИОНИКА» [Электронный ресурс]. – URL: https://rsa.navigat.ru/index.php?id=2 (дата обращения 05.05.2023).
2. Костин А. С. Исследование моделей и методов маршрутизациии практического выполнения автономного движения беспилотными транспортными системами для доставки грузов / А. С. Костин, Н. Н. Майоров // Вестник государственного университета морского и речного флота им. адмирала С.О. Макарова. – 2023. – Т. 15, № 3. – С. 524-536.
3. Yunhong Y.UAV Formation Trajectory Planning Algorithms: A Review / Yunhong Y., Xingzhong X., Yuehao Y. // Drones 2023, 7, 62. – Unmanned Ground and Aerial Vehicles (UGVs-UAVs) for Digital Farming. – С. 4-5.
4. Muñoz J. Multi UAV Coverage Path Planning in Urban Environments / Muñoz J., López B., Quevedo F., A. Monje C., Garrido S. and E. Moreno L. // Sensors – 2021. – 21(21), 7365. – 18 с.
5. Matlab code realization of the shortest path Dijkstra algorithm [Электронный ресурс]. – URL: https://www.programmersought.com/article/85217237737/ (дата обращения 11.05.2023).

ИНФОРМАЦИЯ ОБ АВТОРЕ

Виноградова Екатерина Сергеевна –
студент кафедры эксплуатации и управления аэрокосмическими системами
Санкт-Петербургский государственный университет аэрокосмического приборостроения
190000, Санкт-Петербург, ул. Большая Морская, д. 67, лит. А
E-mail: k2022980@mail.ru

INFORMATION ABOUT THE AUTHOR

Vinogradova Ekaterina Sergeevna –
student of the Department of Operation and Management of Aerospace Systems
Saint-Petersburg State University of Aerospace Instrumentation
67, Bolshaya Morskaia str., Saint-Petersburg, 190000, Russia
E-mail: k2022980@mail.ru