Разработка модификации алгоритма волновой трассировки (алгоритма Ли).
Development of modification of Lee algorithm

Михайлов И. Е.

Михайлов Илья Евгеньевич / Mikhailov Ilya Yevgenyevich – студент,
кафедра компьютерных технологий и систем,
факультет прикладной математики – процессов управления,
Санкт-Петербургский государственный университет, г. Санкт-Петербург

Аннотация: *в данной работе представлен результат исследований по разработке модификации алгоритма волновой трассировки, а также приводится подробное описание алгоритма разработанной модификации вместе с результатами экспериментов, демонстрирующими превосходство в быстродействии созданной модификации над алгоритмом волновой трассировки.*

Ключевые слова: *алгоритм волновой трассировки, алгоритм Ли, алгоритм поиска кратчайшего пути, модификация алгоритма.*

Использование алгоритма волновой трассировки (алгоритма Ли) является одним из способов нахождения кратчайшего пути на графах. Данный алгоритм показал свою эффективность в сравнении с алгоритмом A* [1], поэтому на его основе целесообразно разработать модификацию, которая улучшит показатели алгоритма.

В качестве графа рассматривается равномерная сетка, в которой центры клеток играют роль вершин, при этом соседние клетки выбираются из окрестности Мура (это означает, что соседними клетками считаются 8 клеток по вертикали, по горизонтали и по диагонали) для нахождения ортогонально-диагонального пути. Каждая клетка может являться проходимой или непроходимой.

Модификация алгоритма основана на идее двунаправленного поиска, при котором распространение волны будет происходить не только от начальной клетки, но и от конечной, что позволит обрабатывать меньшее количество вершин графа при поиске кратчайшего пути по сравнению с алгоритмом волновой трассировки. Волну, распространяющуюся от начальной клетки, будем называть прямой, а волну, распространяющуюся от конечной клетки, – встречной.

Главная проблема разработки модификации алгоритма заключалась в создании правила построения кратчайшего пути после встречи волн. Для решения этой проблемы был разработан порядок обхода вершин графа, позволяющий подсчитывать длину кратчайшего пути от каждой клетки, рассмотренной при распространении прямой волны до начальной, и от каждой клетки, просмотренной при распространении встречной волны до конечной [2]. При этом расчет длины происходит немедленно при рассмотрении очередной клетки, не требуя пересчета в дальнейшем. Найденные длины используются для построения искомого кратчайшего пути после встречи волн, что решает вышеупомянутую проблему разработки модификации алгоритма. Для записи и хранения этих длин будем полагать, что каждая клетка имеет атрибут «Длина пути».

Перед началом работы алгоритма будем считать, что атрибуты «Длина пути» у всех клеток пусты, то есть в них ничего не записано. Для удобства будем полагать, что фронт прямой волны, фронт встречной волны, новый фронт прямой волны, новый фронт встречной волны и путь представляют собой структуры данных, аналогичные спискам.

Алгоритм разработанной модификации описан ниже по пунктам.

1. Присвоить атрибутам «Длина пути» начальной и конечной клеток значение 0, добавить эти клетки в новые фронты прямой и встречной волн соответственно.
2. Добавить клетки, содержащиеся в новом фронте прямой волны, во фронт прямой волны в порядке их добавления в новый фронт прямой волны. Удалить все клетки из нового фронта прямой волны.
3. Добавить клетки, содержащиеся в новом фронте встречной волны, во фронт встречной волны в порядке их добавления в новый фронт встречной волны. Удалить все клетки из нового фронта встречной волны.
4. Для каждой клетки, находящейся во фронте прямой волны, в порядке их добавления во фронт прямой волны повторить следующие действия:

а) проверить, не находится ли какой-либо сосед рассматриваемой клетки во фронте встречной волны; если находится, то немедленно перейти к пункту 7;

б) определить таких ортогональных соседей рассматриваемой клетки, для которых одновременно выполняются следующие условия: эти соседи являются проходимыми клетками, и у этих соседей атрибут «Длина пути» является пустым;

в) добавить этих соседей в новый фронт прямой волны;

г) присвоить атрибутам «Длина пути» этих соседей значение атрибута «Длина пути» рассматриваемой клетки, увеличенное на расстояние от рассматриваемой клетки до ортогонального соседа;

д) проделать аналогичные действия с диагональными соседями рассматриваемой клетки, увеличивая значение атрибута «Длина пути» рассматриваемой клетки на расстояние от рассматриваемой клетки до диагонального соседа.

5. Для каждой клетки, находящейся во фронте встречной волны, в порядке их добавления во фронт встречной волны повторить следующие действия:

а) проверить, не находится ли какой-либо сосед рассматриваемой клетки в новом фронте прямой волны; если находится, то немедленно перейти к пункту 9;

б) определить таких ортогональных соседей рассматриваемой клетки, для которых одновременно выполняются следующие условия: эти соседи являются проходимыми клетками, и у этих соседей атрибут «Длина пути» является пустым;

в) добавить этих соседей в новый фронт встречной волны;

г) присвоить атрибутам «Длина пути» этих соседей значение атрибута «Длина пути» рассматриваемой клетки, увеличенное на расстояние от рассматриваемой клетки до ортогонального соседа;

д) проделать аналогичные действия с диагональными соседями рассматриваемой клетки, увеличивая значение атрибута «Длина пути» рассматриваемой клетки на расстояние от рассматриваемой клетки до диагонального соседа.

6. Если новые фронты прямой и встречной волн не являются пустыми – удалить все клетки из фронта прямой волны и из фронта встречной волны, перейти к пункту 2; иначе – завершить выполнение алгоритма, пропустив следующие четыре пункта.

7. Построить путь следующим образом:

а) из всех клеток, находящихся во фронте прямой волны, выбрать те, у которых существует хотя бы один сосед, находящийся во фронте встречной волны;

б) из выбранных клеток найти ту, у которой значение атрибута «Длина пути» минимально, назвать ее центральным узлом;

в) добавить центральный узел в начало пути, сделать его текущей клеткой;

г) найти таких соседей текущей клетки, которые содержались во фронте прямой волны. Выбрать из таких соседей клетку, у которой значение атрибута «Длина пути» минимально, добавить ее в начало пути, сделать ее текущей клеткой;

д) если начальная клетка не добавлена в начало пути – перейти к предыдущему пункту;

е) сделать центральный узел текущей клеткой;

ж) найти таких соседей текущей клетки, которые содержались или содержатся во фронте встречной волны. Выбрать из таких соседей клетку, у которой значение атрибута «Длина пути» минимально, добавить ее в конец пути, сделать ее текущей клеткой;

з) если конечная клетка не добавлена в конец пути – перейти к предыдущему пункту.

8. Вернуть путь в качестве результата работы алгоритма и завершить выполнение алгоритма, пропустив следующие два пункта.

9. Построить путь следующим образом:

а) из всех клеток, находящихся во фронте встречной волны, выбрать те, у которых существует хотя бы один сосед, находящийся в новом фронте прямой волны;

б) из выбранных клеток найти ту, у которой значение атрибута «Длина пути» минимально, назвать ее центральным узлом;

в) добавить центральный узел в начало пути, сделать его текущей клеткой;

г) найти таких соседей текущей клетки, которые содержались или содержатся во фронте прямой волны или содержатся в новом фронте прямой волны. Выбрать из таких соседей клетку, у которой значение атрибута «Длина пути» минимально, добавить ее в начало пути, сделать ее текущей клеткой;

д) если начальная клетка не добавлена в начало пути – перейти к предыдущему пункту;

е) сделать центральный узел текущей клеткой;

ж) найти таких соседей текущей клетки, которые содержались во фронте встречной волны. Выбрать из таких соседей клетку, у которой значение атрибута «Длина пути» минимально, добавить ее в конец пути, сделать ее текущей клеткой;

з) если конечная клетка не добавлена в конец пути – перейти к предыдущему пункту.

10. Вернуть путь в качестве результата работы алгоритма и завершить выполнение алгоритма.

В ходе практического сравнения разработанной модификации с алгоритмом волновой трассировки по быстродействию в различных условиях было установлено, что модификация превосходит алгоритм волновой трассировки в среднем в два раза по скорости работы.

Таким образом, на основе алгоритма волновой трассировки (алгоритма Ли) была разработана его модификация, которая в среднем в два раза превзошла эффективность алгоритма волновой трассировки по быстродействию.

Литература

1. *Михайлов И. Е.* Практическое сравнение алгоритма A* с алгоритмом волновой трассировки (алгоритмом Ли) по быстродействию // Наука, техника и образование. 2016. № 1 (19). С. 47-49.
2. *Михайлов И. Е.* Порядок обхода вершин графа в алгоритме волновой трассировки (алгоритме Ли) // Наука, техника и образование. 2016. № 2 (20). С. 9-11.