В. А. Черненый

АЛГОРИТМЫ РЕШЕНИЯ ЗАДАЧИ БЫСТРОГО ПОИСКА ПУТИ В КОМПЬЮТЕРНЫХ ИГРАХ

В данной работе рассматриваются алгоритмы поиска оптимального пути на игровом поле, способные обеспечить построение траектории движения объекта за оптимальное время с использованием небольших вычислительных мощностей.

Введение

При создании симуляторов, подразумевающих перемещение различных типов объектов по большим территориям с учетом текущей тактической обстановки, возникают проблемы с выбором алгоритма поиска оптимального пути, так как на его использование накладываются ограничения: большой объем данных реальных карт местности, превышающий объем оперативной памяти, сложность представления ландшафта, большой разброс в сложности получаемого пути: оптимальным решением может оказаться как прямая, так и сильно изломанная линия.

I. Классификация алгоритмов

Алгоритмы поиска оптимального пути можно разделить на две группы: алгоритмы позволяющие определить оптимальной путь и алгоритмы позволяющие найти субоптимальный путь. Первой группе требуется полностью исследовать игровую область. Самым простым способом является полный перебор всех возможных марштрутов. Однако такой способ неприменим в большинстве случаев из-за чрезмерных накладных расходов. В связи с этим на первый план выходят алгоритмы поиска субоптимального пути. Примерами являются эвристические алгоритмы, которые на каждом шаге приближаются к конечной точке. [1]

II. Агоритмы поиска оптимального пути

Алгоритм Дейкстры. Этот алгоритм назван по имени создателя и был разработан в 1959 году. В процессе выполнения алгоритм проверит каждую из вершин графа, и найдет кратчайший путь до исходной вершины. Стандартная реализация работает на взвешенном графе — графе, у которого каждый путь имеет вес, т.е. “стоимость”, которую надо будет “заплатить”, чтобы перейти по этому ребру. При этом в стандартной реализации веса неотрицательны. На клетчатом поле вес каждого ребра графа принимается одинаковым (например, единицей).

Алгоритм A*. Данный алгоритм является расширением алгоритма Дейкстры, ускорение работы достигается за счет эвристики — при рассмотрении каждой отдельной вершины переход делается в ту соседнюю вершину, предположительный путь из которой до искомой вершины самый короткий. При этом существует множество различных методов подсчета длины предполагаемого пути из вершины. Результатом работы также будет кратчайший путь.

Поиск по первому наилучшему совпадению (BFS). Усовершенствованная версия алгоритма поиска в ширину, отличающаяся от оригинала тем, что в первую очередь развертываются узлы, путь из которых до конечной вершины предположительно короче. Т.е. за счет эвристики делает для BFS то же, что A* делает для алгоритма Дейкстры.

IDA* (A* с итеративным углублением) Расшифровывается как Iterative Deeping A*. Является измененной версией A*, использующей меньше памяти за счет меньшего количества развертываемых узлов. Работает быстрее A* в случае удачного выбора эвристики. Результат работы — кратчайший путь.

III. Выводы

В работе представлен краткий обзор и классификация алгоритмов поиска оптимального пути. Рассмотрены наиболее эффективные алгоритмы для получения реалистичного пути в случаях, когда затраты на обращение к ландшафту оказываются критичными.

1. What is xAPI aka the Experience API or Tin Can API [Электронный ресурс]. – Режим доступа : https://xapi.com/overview/.
2. Project Tin Can – The Next Generation of SCORM [Электронный ресурс]. – Режим доступа : https://gowithfloat.com/2012/04/project-tin-can-the-next-generation-of-scorm/.

Черненый Владислав Андреевич, магистрант кафедры информационных технологий автоматизированных систем БГУИР, vladislav.chernyony@gmail.com.

Научный руководитель: Шибут Марина Станиславовна, кандидат технических наук, доцент кафедры управления информационными ресурсами академии управления, shibut_ms@pac.by.