INTERNATIONAL JOURNAL OF RESEARCHES IN BIOSCIENCES, AGRICULTURE AND TECHNOLOGY
© VISHWASHANTI MULTIPURPOSE SOCIETY (Global Peace Multipurpose Society) R. No. MH-659/13(N)
www.vmsindia.org

PATH PLANNING FOR HUMAN MOTION USING ASTAR ALGORITHM

Prathama Parag Timbadia and Vivek Kshirsagar
Government College of Engineering,Aurangabad
prathamat84@gmail.com

Abstract
Calculating and generating optimal motion path automatically is one of the key issues in human motion path planning.To solve the problem, Astar algorithm has been analyzed and reviewed so as to improve the search time.Astar algorithm uses an evaluation function to find the best path from source to destination. In today's world, A* algorithm is the most widely used pathfinding algorithm of artificial intelligence and it is also one of the most effective shortest pathfinding algorithms.We put forward deficiency of standard A* algorithm, because standard A* algorithm may detour in the search process and propose an improved A* algorithm making the algorithm more effective and accurate in finding feasible path in unknown environments.
Keywords: A* Algorithm, Improved Astar, Evaluation Function.

Introduction

Nils Nilsson in 1964 suggested a heuristic approach to increase the speed of Dijkstra's Algorithm and called it A1[1].Later Bertram Raphael in 1967 made significant improvements in A1 algorithm and called the version of A1 as A2.But he failed to show optimality .Then Peter E Hart in 1968 introduced an argument that proved A2 was optimal when using consistent heuristic with only minor changes.It showed that the new A2 algorithm was the best possible algorithm given the conditions.He thus named the algorithm A*.

Astar algorithm is one of the most widely used pathfinding algorithms.It has a wide range of applications,may it be in the field of computer networks or robotics,GIS,virtual reality systems, gaming,etc.Basically,Astar algorithm is derived from Dijkstra algorithm [2].We can also say that Astar algorithm is a special case of Dijkstra's Algorithm.Dijkstra's algorithm finds shortest path but wastes time exploring directions.Greedy Best Search algorithm may not find shortest path but explores in promising directions.A* algorithm combines the best features of both the algorithms..Astar algorithm uses evaluation function or heuristic function which makes it faster than the dijkstra's algorithm.

There are two types of search strategies namely non-information-search(or blind search) and information search(heuristic).Blind search algorithm has no information about the search space. [3].The only thing blind search can do is to distinguish non-goal state from goal state.Examples of blind search is breadth first search, depth first search,etc.Information search algorithm has information about the search space.Examples of information search are pattern recognition problems, searching and sorting problems.The A* algorithm is a combination of the best features of informed search algorithm and blind search algorithms.The traditional A* algorithm slows down and easily falls into the failed search state when the obstacles are met in unknown environments. In this paper, we try to propose an improved version of the traditional A* algorithm for path planning.

A* ALGORITHM

In the field of heuristic searching algorithm, A* algorithm is a best-first search algorithm that finds the least cost path from source to destination.The most essential part of A* algorithm is a good estimate of evaluation function(also called as heuristic function).Formula for the evaluation function is given is

$$f(n)=g(n)+h(n)$$

where g(n) is the actual cost from initial node to the current node n.h(n) is the heuristic estimate of distance to the goal.In simple words,it can be defined as path from node n to a goal.h(n) plays a crucial role in finding the optimal path.

To efficiently compute optimal solutions,A* algorithm combines features of pure heuristic search and uniform cost search.If h(n) is admissible,A* algorithm guides an optmal path to a goal.h(n) is admissible implies that the algorithm never overestimates the actual cost.

Different from many other path searching algorithms,it doesn't traverse the entire map and all the nodes.Searching process of A* algorithm is as follows-

1)Mark the initial node and expand the unmarked subsequent nodes(also called as child nodes).

2)Calculate evaluation function value for each subsequent nodes,sort them according to the value of the evaluation function and then

identify and mark the node of the minimum evaluation function value.Stop the search only when the current node is the target node.

The most important step in A* algorithm is to choose the evaluation function.The selection of evaluation function determines whether the algorithm will find the optimal solution or not.

EVALUATION FUNCTION

As mentioned in earlier section,the evaluation function is defined by the formula

$$f(n) = g(n) + h(n)$$

Generally h(n) is evaluated by Euclidean distance formula or manhattan distance formula.Euclidean distance defined by the formula

$$((x_1-x_2)^2+(y_1-y_2)^2)^{1/2}$$

whereas manhattan distance is defined by the formula

$$|x_1-x_2|+|y_1-y_2|$$

Where x_1,x_2 and y_1,y_2 are x and y coordinates of node1 and node2 respectively.

Choosing Euclidean distance between two nodes as h(n) provides the most optimal solution[4].

If $h(n) \leq h^*(n)$(being inclined to a conservative estimate) for all nodes n in the path searching problem,then the best path to reach the target node will be found in A algorithm[5].If lower bound h(n) of h*(n) is looked as heuristic function in A algorithm,then the algorithm is known as A* algorithm.

When the initial node and node n are combined into one node,the above expression becomes

$f^*(n) = h^*(n)$

The above mention function is an estimate of f*(n),.i.e.,where g(n) is an evaluation function of g*(n) and satisfies $g(n) \geq g^*(n)$.Similarly, the h(n) is an estimate of h*(n) and an evaluation function of h*(n) as well.

Weghted processing is done to the valuation function according to the following acceptable lemma[6].

Lemma:There is always a node n* in each step of the A* before termination,which has the following characteeristica on table OPEN:

1) n* is on the best path of achieving the objectives.
2) A* has found the best paths to reach n*.
3) $f(n^*) \leq f(n0)$.

Then the algorithm is acceptable.

There is a corollary to this theory called the Graceful Decay of Admissibility which states that if your heuristic rarely over-estimates the real distance to goal by more than a certain value (lets call it E) then the algorithm will rarely find a solution which costs more than E over the cost of the optimal solution[7].

In our research, we use Chebyshev distance between two nodes as h(n), which is defined as

$$\max(|x_1-x_2|,|y_1-y_2|)$$

and try to find out results for the same.

FLOWCHART ALGORITHM

The Astar Algorithm[8] is as follows

1.Create a search graph G consisting solely of start node n_0. 2.Put n_0 on a list called OPEN.Create a list called CLOSED tht is initially empty.

3. If OPEN is empty, exit with failure.

4. Else, select the first node on OPEN, remove it from OPEN, and put it on CLOSED.Call this node n.

5.If n is a goal, exit successfully with the solution obtained by tracing a path along the pointers from n to n_0 in G(the pointers define a search tree and are established in step 7).

6. Expand node n, generating the set M, of its successors tha are not already ancestors of n in G.Install these members of M as successors of n in G.

7. Establish a pointer to n from each of those members of M that were not already in G(.i.e. not already ancestors of n in G).Add these members of M to OPEN.For each member of M already on CLOSED, redirect the pointers of each of its descendants in G so that they point backwards along the best paths found so far to these descendants.

8.Reorder the list OPEN in order of increasing f values(ties among minimal f values are resolved in favour of the deepest node in the search tree).

9.Go to step 3.

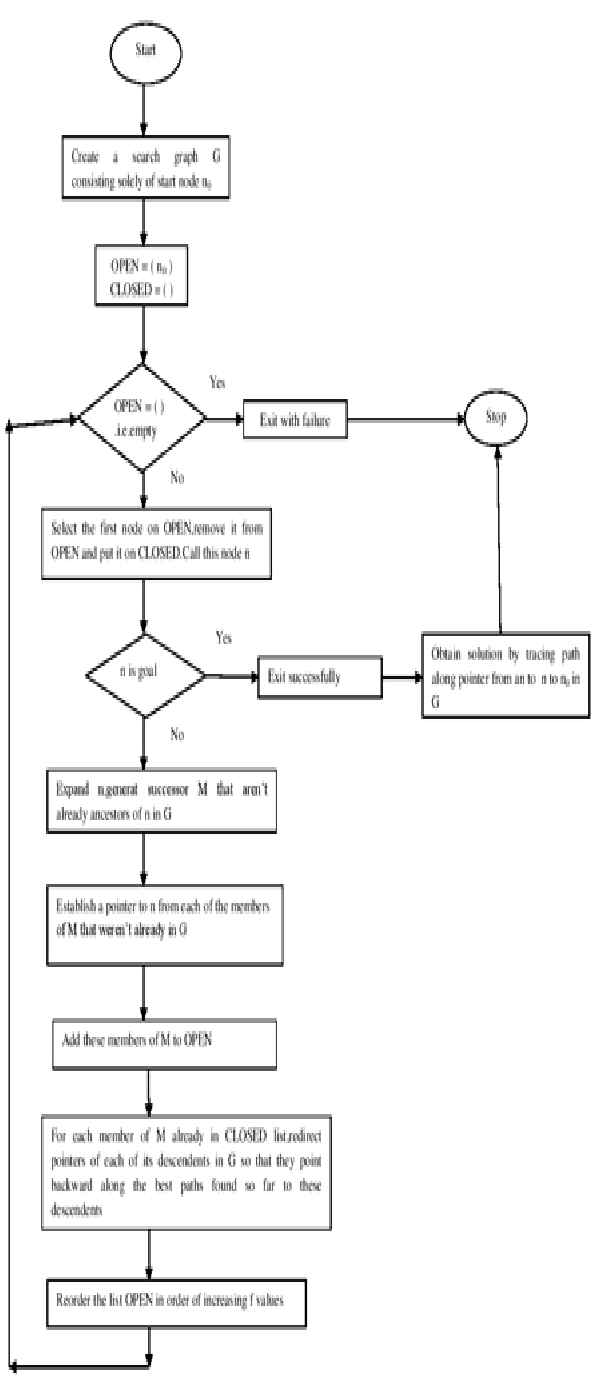

Figure 1: Flowchart of Astar Algorithm

Conclusion

A* algorithm is one the most classical algorithms in aritificial intelligence .It can not only be used in finding shortest path in games,but also in the real world for finding the optimized shortest path between the source and the destination.In this paper, the A* algorithm is applied in path planning of human motion in both known and unknown environment.

At the same time ,the traditional A* algorithm has been improved.The improvement is done in the most critical part of the algorithm .i.e. the evaluation function.The improvement is done in such a way that the improved algorithm not only gives the optimal solution but also finds the solution faster than the traditional A* algorithm.Hence the efficiency of A* algorithm is improved and the effectiveness of the A* algorithm to the unknown environment with obstacles is also enhanced.

References

https://www.princeton.edu/~achaney/tmve/wiki100k/docs/A*_search_algorithm.html

Srikanth Bandi, Daniel Thalmann : "Path finding for human motion in virtual environments"[Z].Electronic Imaging and Media Communications, University of Bradford, Bradford, West Yorkshire, BD7 1DP.

www.cs.nott.ac.uk/~gxk/courses/g5aiai/003blindsearches/blind_searches.htm

Junfeng Yao, Chao Lin, Xiaobiao Xie, Andy JuAn Wang, Chih-Cheng Hung, "Path Planning for Virtual Human Motion Using Improved A* Algorithm" , International Conference on Information Technology, 2010

Mahmoud Tarokh : "Hybrid Intelligent Path Planning for Articulated Rovers in Rough Terrain" [Z].Fuzzy Sets and Systems.2008,159(21):2927-2937.

Gao Qingji, Yu Yongsheng, Hu Dandan, "Feasible Path Search and Optimization Based on an improved A* algorithm", China Civil Aviation College Journal, 2005, 23(4):42-44.

http://heyes-jones.com/astar.php