2009 Third International Conference on Multimedia and Ubiquitous Engineering

A 2-D ESPO Algorithm and Its Application In Pedestrian Path Planning Considering Human Behavior

Zheng Pan[1,3] Lei Yan[1] Adam C. Winstanley[2] A.Stewart Fotheringham[3] Jianghua Zheng[2]

1. Beijing Key Lab of Spatial Information Integration & Its Applications, Institute of RS&GIS, Peking University, Beijing, China

2. Department of Computer Sciences, National University of Ireland, Maynooth, Co.Kildare, Ireland

3. National Center for Geocomputation, National University of Ireland, Maynooth, Co.Kildare, Ireland

***Abstract*—**A 2-D Euclidean shortest path with obstacles (ESPO) algorithm for pedestrian navigation is developed. ESPO is a classical algorithm in the field of computational geometry. We describe some common ESPO algorithms and discuss their application in pedestrian shortest path determination based on the generation of a network of paths within a polygon with interior obstacles. This algorithm can be applied to pedestrian navigation in open spaces, such as squares, parks and big halls. Path generation is based on the Dijkstra algorithm, which is extended to solve the path planning problem not only for path and road networks but also for open spaces. The algorithm takes human preferences based on walking conditions into account and can find different paths with minimum cost for different conditions. The results of this approach are illustrated through an experimental system. Further work to integrate the algorithm into a practical pedestrian navigation system is proposed.

Keywords-ESPO Algorithm; Pedestrian path planning; Pedestrian behavior

I. INTRODUCTION

It is not always easy for pedestrians to find the right way to reach their destination in an unfamiliar environment. Pedestrian path planning algorithms can help them to find an optimal route. However, most of today's existing path planning algorithms are designed for vehicle navigation and therefore are not ideally suited for pedestrian applications.

Road networks for vehicle navigation only include those parts that are accessible and permissible for vehicles. Pedestrians are not tied to these roads, for example, they can cross open spaces instead of following a constrained path. Therefore, the biggest difference between vehicle networks and pedestrian networks is the latter include open spaces or walking areas.

Walking areas can be defined as features that are accessible open areas within which no defined path network is contained. Walking areas includes grasslands, parks, squares, woods, large halls and any other area without an obvious path network. Sometimes, there may be internal features such as a pond or a lawn which can not be traversed. Pedestrians need to avoid these obstacles.

In this paper, we focus on the task of how to find the shortest path for a pedestrian in a walking area and, at the same time, how to avoid internal obstacles. In section 2, we introduce the graphical representation of walking areas and explain why the problem of finding a path can be reduced to an ESPO problem. A detailed description of the problem is also given. Following that, in section 3, we introduce related work in the field of ESPO algorithms and two main methods used to solve the problem. In section 4, we present a pedestrian shortest path algorithm for walking areas, taking into consideration human preferences in different conditions. Simulated results of using the algorithm are given. The paper closes with some concluding remarks and suggestions of enhancements for a practical pedestrian navigation system.

II. PROBLEM DESCRIPTION

A. Graphical representation of walking areas

In common road network topologies, we convert the real road network into an aggregation of nodes and links. In a pedestrian road network, walking areas are also important features. They are usually connected into a path network at specific entrance points.

We can represent a walking area by its minimum bounding polygon, which can be convex or concave. Polygons are defined by a boundary of straight lines, and the shape is 'closed' (all the lines connect up). For example, Figure 1 shows a polygon representing a recreation field, which is an open area that can be walked across by pedestrians. In this figure, blue lines represent boundaries of the walking area and yellow points represent vertices along the boundary.

Figure 1. Using a polygon to represent a walking area

978-0-7695-3658-3/09 $25.00 © 2009 IEEE
DOI 10.1109/MUE.2009.86

Sometimes, there may be an obstacle which can not be traversed. Pedestrians need to avoid these obstacles. We can also use its minimum bounding polygon to represent the obstacle and use a polygon with 'island' to represent the walking area, as shown in the Figure 2.

To be specific, we use simple polygons to represent walking areas. A simple polygon is a polygon whose sides do not intersect. Such a polygon divides the plane into two regions - the region inside and region outside. A polygon that is not simple is said to be self-intersecting. Such a polygon does not necessarily have a well-defined interior and exterior. When path planning for pedestrians, we need to know if a location is inside a polygon, so the simple polygon is an appropriate topology type. Self-intersecting polygons can always be divided into several simple polygons. As shown in Fig. 3, the polygon representing walking area A is a self-intersecting polygon, which intersects itself at point S. This polygon can be divided into two simple polygons A1 and A2 and the intersection point S can be defined as a vertex.

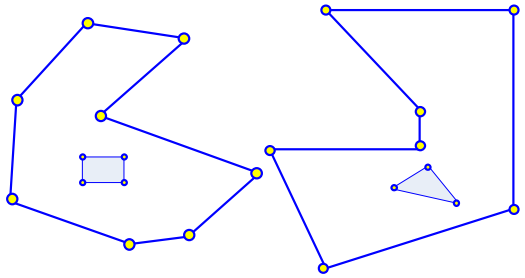

Figure 2. Polygons with islands

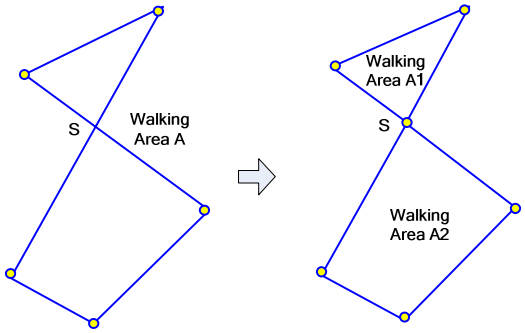

Figure 3. Converting a self-intersecting polygon into a simple polygon

Therefore, polygons mentioned in the following content are all simple polygons. The problem of pedestrian path planning across a walking area can be defined as finding a shortest path inside a simple polygon.

B. Problem description

In most cases, the shortest path inside a simple polygon may be the straight line from the start point to end point, as shown in the Figure 4.

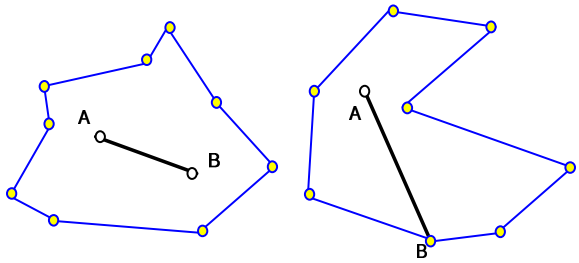

Figure 4. Straight shortest paths inside polygons

Sometimes the shortest path may have to go around obstacles. With concave polygons, the shortest path will often pass by vertexes or hug the edge of the polygon for part of its journey, as shown in the Figure 5.

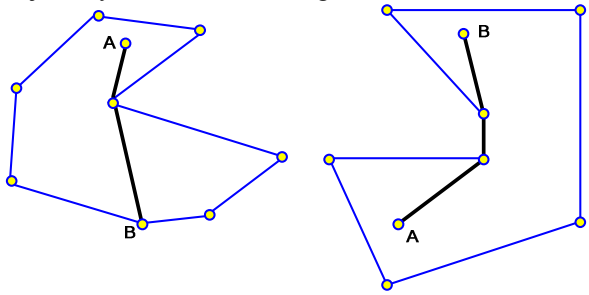

Figure 5. Shortest paths in concave polygons

Sometimes the shortest path needs to avoid internal obstacles, as shown in Figure 6.

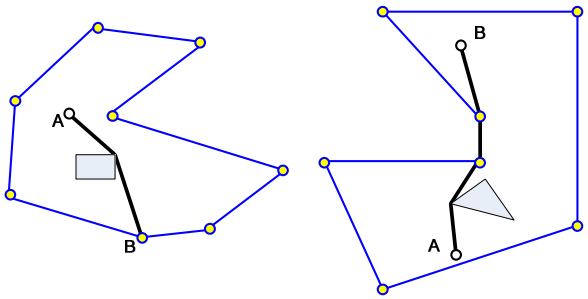

Figure 6. Shortest paths avoiding internal obstacles

Therefore, the abstract description of the problem of finding the shortest path inside walking areas can be defined as: given a simple polygon and a set of polyhedral obstacles in the polygon, a start point and an end point within the polygon, find the shortest path inside the polygon between the two points that avoids all of the obstacles.

According to this description, this problem belongs to the ESPO (Euclidean shortest path with obstacles) problem. In this paper, we regard a walking area as a two dimensional plane, and the algorithm to solve the problem is a 2-D ESPO algorithm.

III. 2-D ESPO ALGORITHMS

The ESPO problem is well-known in computational geometry: given a set of polyhedral obstacles in a Euclidean plane, and a pair of locations, find the shortest path between the points that does not intersect any of the obstacles. In two dimensions, the ESPO problem can be solved efficiently, in polynomial time. In three (and higher) dimensions, however, the problem is NP-hard in the general case.

Although the problem has drawn the attention of many researchers in computational geometry, researchers in geography pay little attention to it because existing path planning algorithms are usually based on linear graph networks rather than planes. According to the different application fields, there are two main kinds of solutions of the problem: geometric methods and geographic methods.

A. Geometric method: Triangulation

In the field of geometry, this problem have been researched for many years and several algorithms have been proposed to solve it [1] [2] [3]. Most of the methods are based on a triangulation of the polygon. Lee and Preparata's approach finds the shortest path between two points inside a simple polygon in linear time, once a triangulation is known [4]. The algorithm of Reif and Store [5] uses pre-computation to speed up queries. Given a source point inside the polygon, their method produces a search structure so that the distance from the source point to a query point can be found in O(log n) time. The shortest path itself can be obtained in time proportional to the number of turns along it. Reif and Storer's method uses the Delaunay triangulation of the polygon and hence takes O(n log n) preprocessing time. Guibas et al. [6] show how to set up a similar query structure with less preprocessing. Their algorithm takes linear time once a triangulation is known.

B. Geographic method: Extended shortest path algorithm

In geographic information and navigation systems, shortest path algorithms are central to many tasks. For a given source node in a pedestrian road network, this problem is to find the path to the end node with the lowest cost. There are many well-known methods such as Dijkstra's algorithm and Floyd's algorithm [7] [8]. Floyd's algorithm is used in calculating the shortest path among all the nodes, while Dijkstra's algorithm is used in calculating the shortest path from one node to all other nodes. However, these algorithms are usually used to deal with a graph with nodes and lines and can't be directly used to find a path in open space. Some researchers have proposed extended algorithms to solve ESPO problems, which are based on the classical shortest path algorithms, such as Dijkstra or Floyd algorithm but there are not detailed descriptions of these algorithms.

This paper proposes an extended Dijkstra algorithm to solve the problem of shortest path finding in walking areas.

IV. PEDESTRIAN SHORTEST PATH ALGORITHM INSIDE WALKING AREAS

A. Description of algorithm

Let G= (V, E) denote a polygon in the plane having h holes ("obstacles") and a total of n vertexes. V is a set of vertexes and V= $\{v_1, v_2, \ldots, v_n\}$, including vertexes of the polygon and vertexes of each obstacle inside the polygon. E is a set of edges, including edges of the polygon and edges of each obstacle inside the polygon. Given a start point s $\in$G and an end point t $\in$G, the problem is to find a shortest path inside the polygon from s to t. Fig. 7~Fig. 10 is an example of finding the shortest path.

The main steps of the algorithm are as follows:

a) Given a polygon and a pair of points, as shown in Figure 7

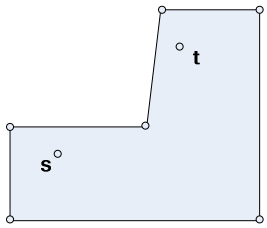

Figure 7. Polygon and start/end point

b) Connect the start point s to each vertex of the polygon

c) If the connecting line is completely within the polygon, then take it as a link, as shown in Figure 8.

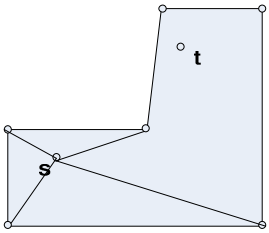

Figure 8. Connecting start point and polygon vertices

d) Execute the same steps to the end point, as shown in Figure 9.

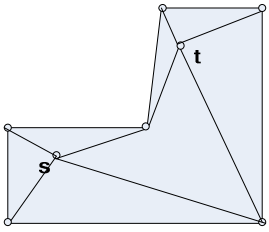

Figure 9. Connecting end point and polygon vertices

e) Connect each pair of vertices of the polygon

f) If the connecting line is in the polygon, then take it as a link, as shown in Figure 10.

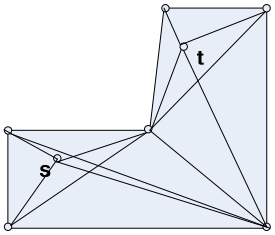

Figure 10. Connecting pairs of vertices

g) Form a path network using all polygon edges and links
h) Find a shortest path in the road network using Dijkstra's algorithm, as shown by the dotted lines in Figure 11.

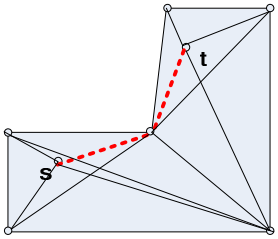

Figure 11. Shortest path from s to t

Fig. 12 is the detail flow chart of our pedestrian shortest path algorithm in walking areas.

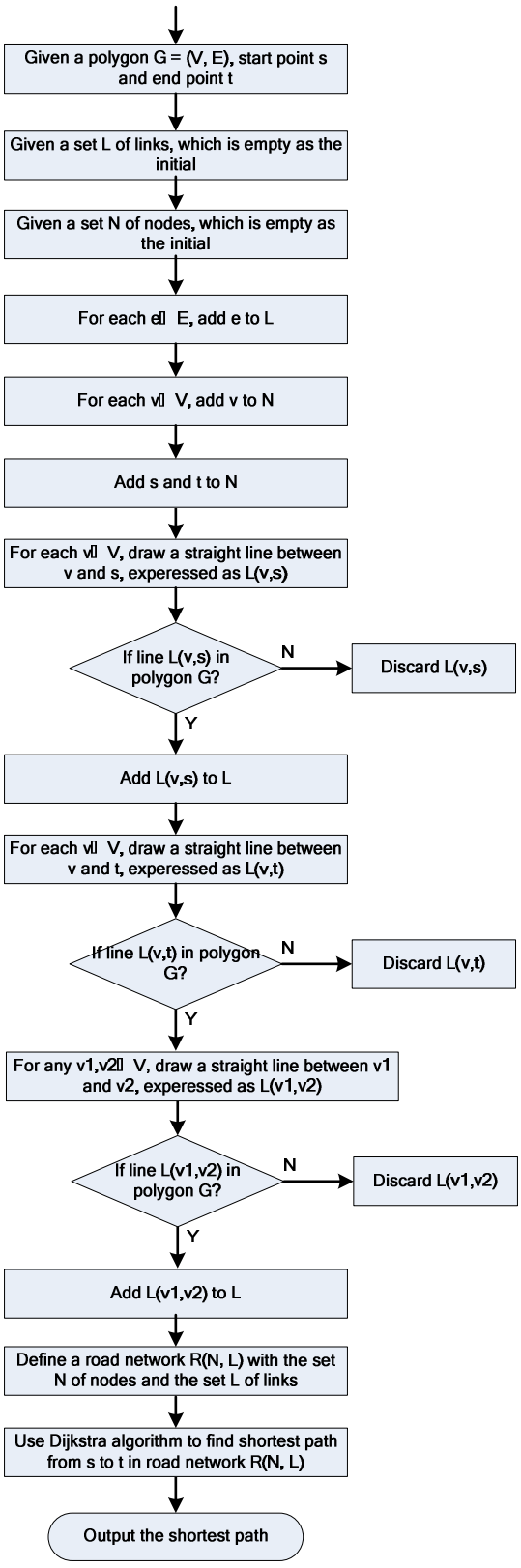

Figure 12. Flow chart for shortest path algorithm in walking areas

B. *Experimental Results*

We take the map of the NUI Maynooth campus as a case study. There are several plots of lawn in the campus, which can be regarded as walking areas. In Figure 13 a pedestrian wants to walk between the points marked by flags.

Figure 13. Example of walking area and start/end points

As mentioned in Figure 12, for each vertex v of the polygon, draw a straight line between v and the start point s, and add these lines to the set L of links by eliminating those lines extending outside the polygon. The result is represented by the black lines shown in Figure 14.

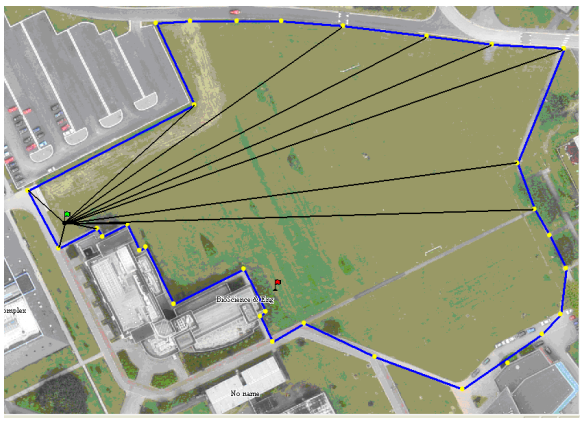

Figure 14. Construct links between start point and vertices

In the same way, connect lines between the end point and all appropriate vertices, and add them to L, shown by the black lines in Figure 15.

Next, connect each pair of vertices of the polygon and add these lines to the set L of links after eliminating those lines extending outside the polygon (Figure 16).

Therefore, the set of links, L, is made up of four parts: edges of polygon (blue lines in Figure 13), links constructed by start point (Figure 14), links constructed by end point (Figure 15), and links constructed by pairs of vertices (Figure 16). The set of nodes, N, is made up of two parts: vertexes of polygon and a pair of locations.

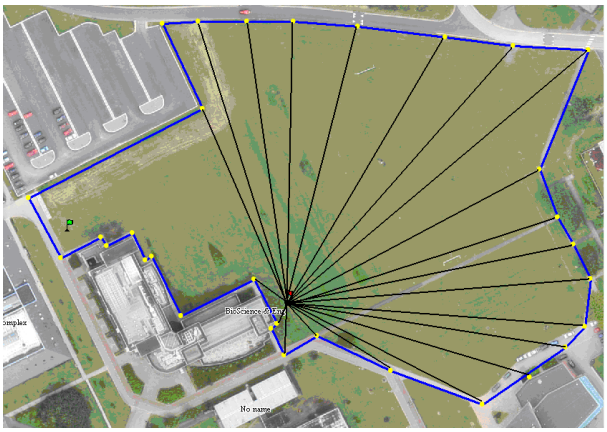

Figure 15. Construct links between end point and vertices

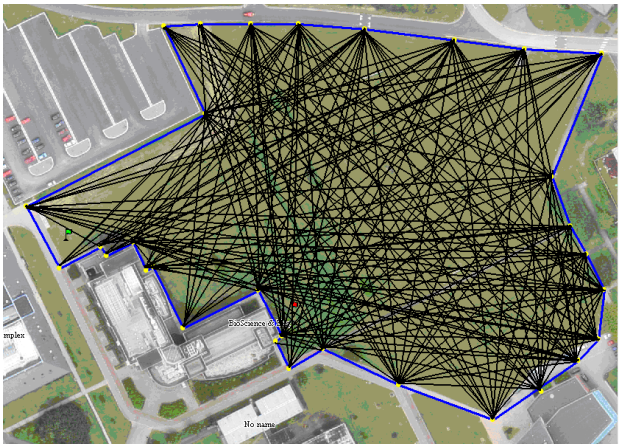

Figure 16. Construct links between each apir of vertices

We can execute Dijkstra algorithm to find the shortest path inside the polygon based on the network R(N, L), as shown in Figure 17.

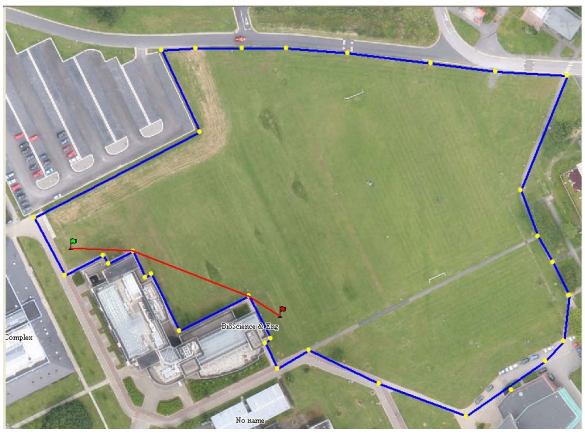

Figure 17. Shortest path computed in walking area

Figure 18 is another example. In this case, the pedestrian would first go straight to a vertex of the obstacle, and walk along an edge of the obstacle, before going to another vertex.

Figure 18. Shortest path following the edge of an obstacle

Taking the network computed in the walking area and the fixed path network on its exterior, we can construct on-the-fly an integrated network, which including nodes, links, and walking areas. A hybrid pedestrian shortest path algorithm can then be applied in the integrated network. By considering human walking preferences, we can apply costs to the network links to develop a practical pedestrian navigation system.

C. *Consideration of Human Behavior*

For pedestrian path planning, different conditions underfoot may influence path selection. A shortest path means a minimum cost path, but the cost used can be path length, journey time, human effort, some other factor or a combination of these. Other factors can be a preference for indoor routes, outdoor routes or paved paths depending on weather conditions. The pedestrian shortest path algorithm can adapt in different conditions by calculating the corresponding costs.

As an example, we will take a path from the main entrance of our campus to a student residence as an example, to show different minimum cost paths.

Shortest Length Path

Under ordinary circumstances, a pedestrian would like to choose the shortest length path. The fundamental link cost is calculated according to the length of each link.

Costs can be varied by changing the cost change rate α, β, as follows:

$$\text{Cost} = \text{length} \bullet \alpha + \beta \qquad (1)$$

In this case, we can set α=1 and β=0.

As shown in Figure 19, the pedestrian takes a shortest length path by crossing grasslands.

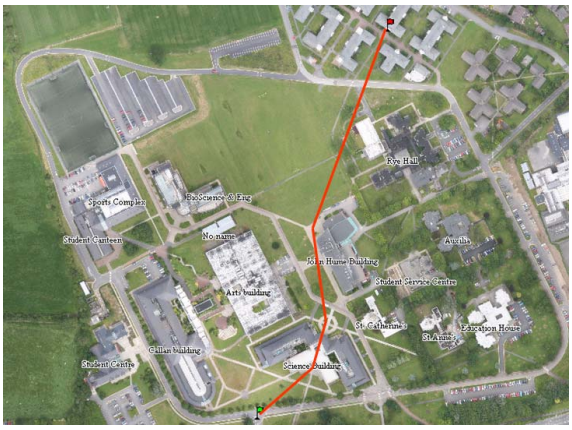

Figure 19. Shortest length path over a hybrid network

Easy-Walk Priority Path

Not all pedestrians demand the shortest path to a destination. For example, wheelchair-bound people would prefer to select a paved road or in wet weather people may prefer to avoid soft surfaces. In these cases, we can choose appropriate values of α and β to reduce the cost of paved roads. As shown in Fig. 20, the pedestrian walk along paved roads to his destination. Although it is not a shortest length path, it's a minimum effort path.

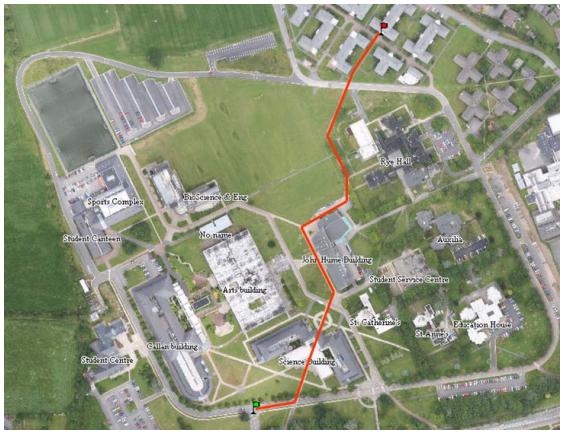

Figure 20. Easy-walk route keeping to paved pathways

Indoor Priority Path

In the case of bad weather, the use of covered sidewalks or indoor routes may be preferred. Accordingly, we can change the value of α and β to reduce the cost of indoor paths (Figure 21).

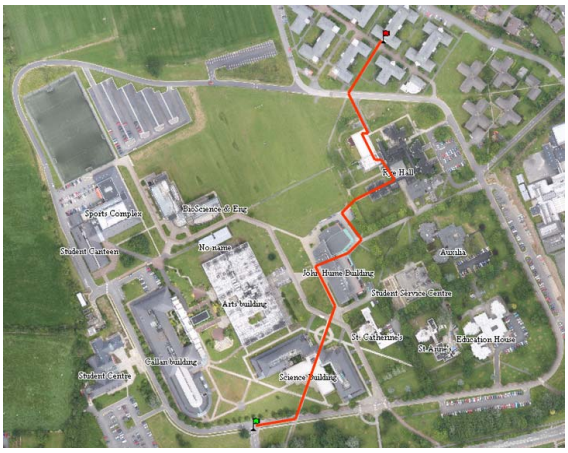

Figure 21. Path with preference for indoor routes

V. CONCLUSION

The Euclidean Shortest Path with Obstacles (ESPO) algorithm can be used to find routes in unrestricted walking areas, such as parks, squares and woods. This paper describes the well-known geometric ESPO algorithm and its application in the field of pedestrian navigation, and proposes a 2-D ESPO algorithm based on the classical Dijkstra algorithm. This algorithm is applicable in open spaces, within which no obvious path network is contained. To make the resulting path suitable for a practical pedestrian navigation system, this algorithm can also consider different human preferences under different environmental conditions by adjusting the costs calculated for each link. In addition it can be integrated into a more classical network-based route-finding algorithm which considers sidewalks, footpaths, foot-bridges, building interiors and other pedestrian routes. Further work will explore this integration and also the communication of routes in cartographic, diagrammatic and textual form for users.

ACKNOWLEDGEMENTS

This project was partly supported by the StratAG project, funded under the Science Foundation Ireland Strategic Research Cluster Programme (07/SRC/I1168).

REFERENCES

[1] Leonidas J. Guibas, John Hershberger, “Optimal shortest path queries in a simple polygon”, Third Annual Symposium on Computational Geometry, pp50 – 63, Waterloo, 1987.

[2] Leo Guibas, John Hershberger, Daniel Leven, Micha Sharir, Rober E. Tarjan, “Linear time algorithms for visibility and shortest path problems inside simple polygons”, 1986

[3] John Hershberger, Subhash Suri, “Efficient computation of Euclidean shortest paths in the plane”, 1993

[4] D. T. Lee and F. Preparata, “Euclidean shortest paths in the presence of rectilinear barriers”, Networks, vol.14 (1984), pp.393-410

[5] J. Reif and J. Storer, “Shortest paths in Euclidean space with polyhedral obstacles”, Technical Report CS-85-121, Computer Science Department, Brandeis University, 1985. Submitted to Journal of the ACM

[6] L. Guibas, J. Hershberger, D. Leven, M. Sharir, R. Tarjan, “Linear time algorithms for visibility and shortest path problems inside simplepolygons”, Proceedings of the 2nd ACM Symposium on Computational Geometry(1986), pp.1-13

[7] Chai Dengfeng, Zhang Dengrong, “Algorithm and its application of N shortest paths problem”, 2001

[8] Nuno M. Garcia, Przemyslaw Lenkiewicz, Mario M. Freire, Paulo P. Monteiro, “On the performance of shortest path routing algorithms for modeling and simulation of static source routed networks – an extension to the Dijkstra algorithm”, 2007