International Journal of Hybrid Information Technology
Vol. 10, No.3 (2017), pp. 21-32
http://dx.doi.org/10.14257/ijhit.2017.10.3.03

An Approach for Generating Pedestrian Network Based on Improved Navigation Graph

Yuelu Chen, Gang Wu, Tianzhong Zhao* and Feixiang Chen

School of Information Science & Technology, Beijing Forestry University,
Beijing 100083, China
**ztz@bjfu.edu.cn*

Abstract

Pedestrian network is the basic data for route planning in pedestrian navigation systems. The optimal route computed by the pedestrian network can be used as wayfinding cues and provide pedestrians with route guidance. Currently, pedestrian network generation methods do not take into account the randomness of walking behaviors, cannot generate the off-road route segments and provide accurate route guidance. In this study, we proposed an approach for generating pedestrian network based on improved navigation graph. First, walking environment was regarded as the space with obstacles. The generation method of navigation points was improved to process the global concave vertices in polygon obstacles. Then, the optimization method of navigation points based on mergence area was proposed to effectively reduce the number of navigation points. Finally, the generation algorithm of navigation edges based on the dynamic cone-shaped sweep was proposed to generate sparse navigation edges. The experiment results showed that this approach can generate the improved navigation graph which data size can be largely reduced. The pedestrian network represented by the improved navigation graph can provide pedestrians with accurate route guidance. This approach can effectively solve the modelling problem of the pedestrian network in pedestrian navigation applications, having a certain practical value.

Keywords: *Pedestrian navigation system, pedestrian network generation, space of obstacles, visibility graph, improved navigation graph, route guidance*

1. Introduction

Pedestrian network is a kind of topological map consists of the geometric relationship between pedestrian route segments, which has been widely used in a variety of fields such as pedestrian navigation and urban planning [1]. In pedestrian navigation systems, pedestrian network is the basic data for route planning and guidance. Currently, pedestrian navigation applications still use road network. However, the differences in scale and details mean that the road network could not replace the pedestrian network [1, 2].

Methods for modelling pedestrian network include manual digitization [3], network buffering based on road centerlines [1], collaborative mapping based on GPS traces [4] and image processing based on LiDAR [5]. These methods are mainly used to extract real roads, such as sidewalk, crosswalk and footpath. However, walking is highly casual and not restricted by regular roads [6], so the pedestrian network generated by these methods could not provide off-road route guidance. The landmark-based oriented visibility graph uses landmarks as the route nodes and generates route segments based on the visible relations between landmarks without dependence on the real road [7-9]. Although this method could adapt to various walking environment, it ignores the area without landmarks.

*Corresponding Author

ISSN: 1738-9968 IJHIT
Copyright ⓒ 2017 SERSC

Wayfinding is a kind of obstacle avoiding behavior based on the interaction between pedestrians and walking environment [10]. Therefore, walking environment could be regarded as two-dimensional Euclidean space with obstacles. The environment modelling methods of the space with obstacles can be categorized into three types: Voronoi diagram [11], Delaunay triangulation [12] and visibility graph [13]. The first two types of methods generate nodes of the network far from the obstacle corners, leading to a lot of invalid route length, unable to reflect the walking habits of the pedestrians. The pedestrians usually walk toward the target direction and make decisions at the corners of obstacles. Hence, the visibility graph could truly and accurately reflect the pedestrians' feeling and cognition in the process of wayfinding [14]. However, with increasing number of obstacles, the number of nodes and edges in the graph would be quite large, leading to low computational efficiency [15]. Although navigation graph [13], tangent graph [16], dynamic visibility graph [17] or simplified visibility graph [18] could reduce the problem space, all of these methods rely on the given route origin and destinations. The graphs generated by these methods cannot cover the whole spatial area and be served as the pedestrian network.

Navigation graph is a microscopic pedestrian simulation model proposed by Kneidl A. and Bormann A. *et al*., [13]. It can be improved to cover the whole spatial area and then applied in the pedestrian navigation systems. This study proposes an approach for generating pedestrian network based on Improved Navigation Graph (ING). Using the space with obstacles to describe the walking environment, the generation method of navigation points is improved. Then, we propose the navigation point optimization method based on mergence area and navigation edge generation algorithm based on Dynamic Cone-shaped Sweep (DCS) to generate the pedestrian network denoted by the improved navigation graph.

2. Improved Navigation Graph

In this study, we regard the spatial obstacles in the real world as two-dimensional space with obstacles represented by planar geometric figures. The spatial area excluding the coverage of obstacles is the walking environment. Space obstacles are categorized into two types: (1) linear obstacles, such as rails and walls; (2) polygonal obstacles, such as buildings and large green belts.

The improved navigation graph is a kind of topological map based on space with obstacles and generated by the global structure procedures of the navigation graph. It is denoted as *ING*= (*NP*, *NE*). Wherein, *NP* and *NE* represent a non-empty set of navigation points and navigation edges, respectively. As a series of wayfinding cues, the pedestrian routes composing of navigation points and edges will provide pedestrians with route guidance. As the generation process does not rely on the Origin and Destinations (OD), the improved navigation graph is able to cover the whole spatial area. Therefore, in this study, we denote the pedestrian network using the improved navigation graph, with the navigation points being the route nodes and the navigation edges being the route segments.

The improved navigation graph has two critical properties: the number of navigation points and edges, respectively denoted as *N* and *E*. *N* and *E* determine data size of the improved navigation graph. Complete Navigation Graph (CNG) is a kind of special visibility graph which contains all navigation points and its adjacency relations [13, 14]. Although the pedestrian route computed by the CNG is able to provide accurate route guidance, the large numbers of *N* and *E* will lead to low computational efficiency. The improved navigation graph could reduce *N* and *E* in the generation process, has less navigation points and edges. But the route computed by ING might detour unnecessary areas, thus leading to inaccurate or even wrong route guidance. Route length is a key factor effecting pedestrian's wayfinding behavior [12,19], as is shown in Figure 1, R_1 and

R_2 are respectively the shortest route computed by the same OD pair using the CNG and ING as the pedestrian network. As the CNG contains all navigation points and edges, R_1 is the shortest route between O and D. Comparatively, the ING ignores the nodes v_1, v_7 and route segments between v_1 and v_7, v_3 and v_6, causing that R_2 needs to pass v_2, v_4, v_5 and v_8. Therefore, the improved navigation graph should be able to compute as accurate as possible routes to provide guidance for the pedestrians with less N and E.

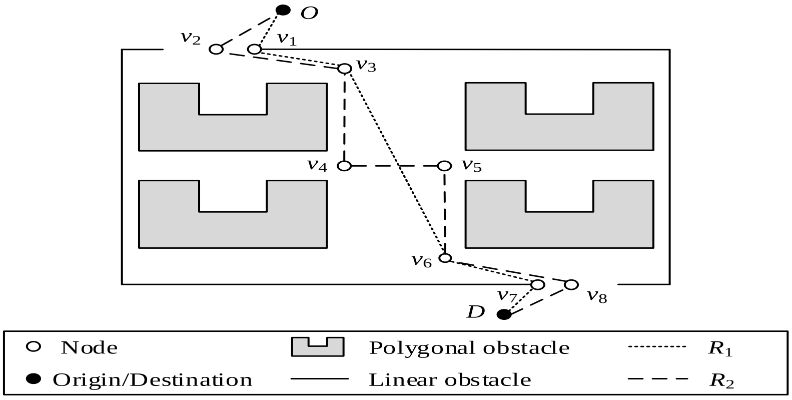


Figure 1. Comparison of Two Routes Computed by CNG and ING between Same OD Pair

3. Generation and Optimization of Navigation Points

3.1. Improved Generation Method of Navigation Points

According to the geometric shapes, we categorized the spatial obstacles into four types: (1) line obstacles; (2) polyline obstacles; (3) polygon obstacles and (4) complicated boundary obstacles. Different methods are used to generate navigation points. To make description convenient, in this study, the larger included angle formed by two adjacent edges in the polyline obstacles is called as the convex angle, the vertices of inner angles less than π as the convex vertices and the vertices of inner angles equal to or more than π as concave vertices. Obstacle vertices generating navigation points are called corresponding vertices. To avoid causing discomfort to pedestrians [12], the nodes should maintain a certain distance with the obstacles. We introduce parameter h to control the distance between navigation points and the corresponding vertices.

For line obstacles, two navigation points are respectively generated on the extended section of the two endpoints. For polyline obstacles, navigation points are generated at the extended sections of the two endpoints and the convex angle's bisector. For polygon obstacles, navigation graph generates navigation points at all of convex vertices [13]. However, as is shown in Figure 2(a), local convex vertices may not global convex vertices. Although V_1 is a convex vertex, it will become a concave vertex if ignoring concave vertices V_2 and V_3. In this study, vertices like V_1 are called global concave vertices. Redundant navigation points generated at the global concave vertices will lead to a lot of invalid navigation edges. Thus, we propose an algorithm to process the global concave vertices:

(1) This algorithm, firstly, searches the vertices of the polygon obstacles in sequence to determine whether its property is concave or convex. Mark the concave vertex. Then the last unmarked vertex is returned to continue the searching, until there are no unmarked concave vertices in the polygon. A formula is used to determine the property of the vertices:

$$C\left(V_i\right)=\left(x_{i+1}-x_{i-1}\right)\left(y_i-y_{i-1}\right)-\left(x_i-x_{i-1}\right)\left(y_{i+1}-y_{i-1}\right) \tag{1}$$

where (x_i, y_i) is the coordinates of vertex V_i. (x_{i-1}, y_{i-1}) and (x_{i+1}, y_{i+1}) respectively represent two adjacent vertices of V_i. When $C(V_i)<0$, V_i is a concave vertex.

(2) For the polygon consisting of unmarked vertex sequences, the vertices are marked between intersect points in two edges, then repeat step (1). If there are no intersected edges, navigation points are generated at the external angular bisection of the unmarked vertices. Figure 2(b) shows the generated navigation points after processing by this algorithm.

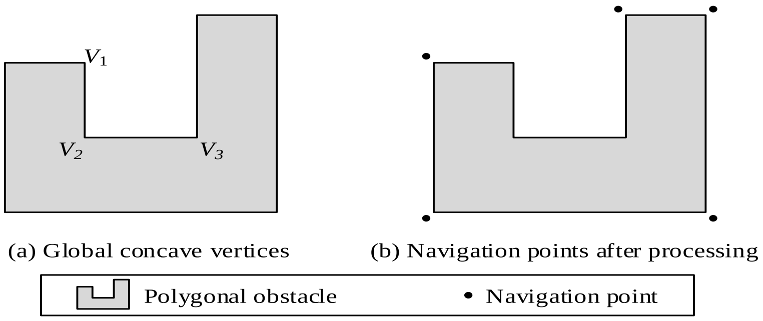

V_1
V_2
V_3
(a) Global concave vertices
(b) Navigation points after processing
Polygonal obstacle
● Navigation point

Figure 2. Gereation Results of Navigtion Points after Processing

For complicated boundary obstacles with curved edges, navigation points are generated on the external bisection of the minimum bounding rectangle of the obstacles. In the process of generating navigation points, the IDs of the corresponding obstacles and the indexes of the corresponding vertices in the obstacles are recorded.

3.2. Optimization Method of Navigation Points

Firstly, this method revise the position of the navigation points to make sure that these points are visible for their corresponding vertices. Although the corresponding angles judgment could merge part of the navigation points [13], this method is restricted regarding the following aspects: (1) It is not able to process the status with close navigation points far away from the corresponding vertices; (2) It could not merge multiple navigation points at the same time; (3) This method does not consider the impact of mergence on the topological relations among navigation points.

In this study, we introduce threshold P to control the maximum distance of mergence between navigation points. The mergence area is defined as per different types of navigation points to avoid dramatic position changes of navigation points after mergence and ensure that two adjacent navigation points of the same obstacle are visible. The mergence area of the navigation point NP is a space area with V (the corresponding vertex of NP) being the endpoint and half-lines VA, VB and their included angle $\angle AVB$. As is shown in Figure 3, Type I is the mergence area of the navigation points of the line obstacles. VA is the extension line of the line obstacle at the direction of end point V. VB is the normal line of end point V at the NP side. Type II is the mergence area of the head and tail end points of the polyline obstacle. VA is the extension line from navigation point NP_0 and V; in addition, NP_0 is the front navigation point of V. VB is the symmetric half-line of VA with VNP being the symmetry axis. Type III is the mergence area of the inflection vertices of the polyline obstacle and the vertices of the polygon obstacle, with VA and VB respectively being the extension lines at the vertex direction of the two edges housing the vertex. Then, we propose an optimization algorithm to merge multiple adjacency navigation points. This algorithm is as follows:

(1) This algorithm, firstly, searches all navigation points. If the current point is not merged, a set of mergence area is initialized and the mergence area of the current point is added into the set.

Copyright © 2017 SERSC

(2) We regard the navigation points which IDs are larger than the current point as the judgment points. If the distance between the current point and the judgment point is shorter than *P*, the median point coordinates equation is used to calculate the new coordinates of the merged navigation point:

$$\begin{cases} x_m = \dfrac{x_c + x_j}{2} \\ y_m = \dfrac{y_c + y_j}{2} \end{cases} \tag{2}$$

where (x_c,y_c) is the current point coordinates, (x_j,y_j) and (x_m,y_m) are the coordinates of the judgment point and the merged point, respectively.

(3) If the merged navigation point locate in the judgment point's mergence area and all mergence areas in the set, mark the judgment point and add its mergence area into the set, and then update the current point coordinates to the merged navigation point.

Repeat these steps until all navigation points are searched. Finally, the unmarked navigation points are the optimization results.

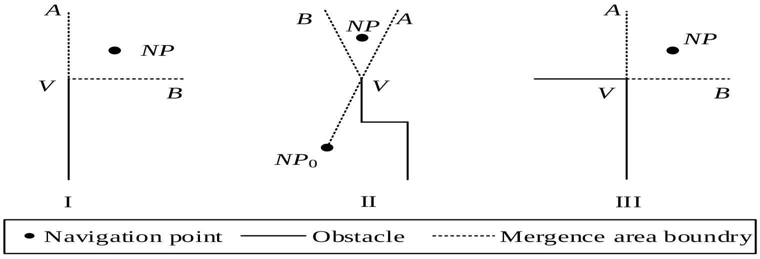


Figure 3. Three Types of Mergence Areas

4. Generation of Navigation Edges

4.1. Dynamic Cone-shaped Sweep

In order to make the improved navigation graph cover the whole spatial area and further reduce the number of navigation edges, we propose a generation method of navigation edges based on the Dynamic Cone-shaped Sweep (DCS). With a vertex being the center, other vertices are used to establish an alterable cone-shaped space. The vertex closest to the central vertex of the space is recorded and others are ignored. In addition, we introduce a sweep angle threshold to control the cone-shaped scope. The higher threshold value will cause the fewer cone-shape and recorded vertices. In this way, we can effectively control the number of generated navigation edges.

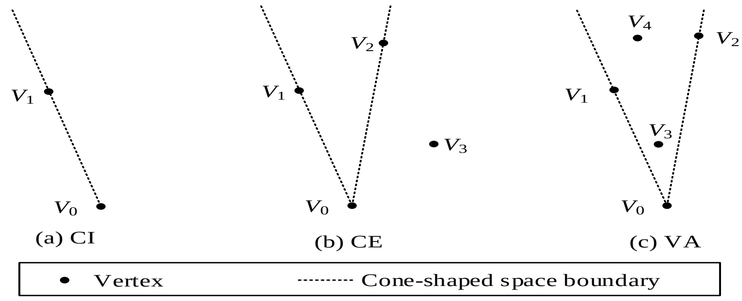
In this study, *Cone* (V_m,V_n) represents the cone-shaped space whose central vertex is *V* and which is formed by the vectors $\overrightarrow{VV}_m$, $\overrightarrow{VV}_n$ and cone angle $\angle V_m V V_n$. *S* denotes the vertex in the cone-shaped space closest to *V*. *D* indicates the threshold of the sweep angle. As is shown in Figure 4, we propose three cone-shaped sweep operations to make the sweep scope more flexible and effective to avoid overlap:

(1) Cone Initialization (CI): Establish an initial cone-shaped space whose included angle is 0° based on the central vertex and any vertex. Furthermore, *S* is stored to the central vertex. As is shown in Figure 4(a), for vertex V_0, an initial cone-shaped space *Cone* (V_1, V_1) is established consisting of V_1. Set *S* to V_1.

(2) Cone Expansion (CE): Expand the scope of the cone-shaped space from one side so that another vertex can be included. Make a judgment whether *S* need to be updated. As is shown in Figure 4(b), as $\angle V_1V_0V_2<D$, *Cone*(V_1,V_1) is expanded to be *Cone*(V_1,V_2) so as

to include V_2. As $|V_0V_1|<|V_0V_2|$, S is set to V_1. In addition, as $\angle V_1V_0V_3>D$, *Cone*(V_1,V_2) cannot be expanded to *Cone*(V_1,V_3) and include V_3.

(3) Vertex Absorption (VA): Absorb the vertices within the sweep scope of the cone-shaped space. The scope of the cone-shaped space remains the same. Make judgment if S need to be updated. As is shown in Figure 4(c), V_3 is in the cone-shaped space *Cone* (V_1, V_2). As $|V_0V_3|<|V_0V_1|$, S is updated to V_3. V_4 is also located within *Cone* (V_1, V_2), but $|V_0V_3|<|V_0V_4|$, there is no need to update S.



(a) CI (b) CE (c) VA

Figure 4. Three Operations of DCS

4.2. Generation Algorithm of Navigation Edges

The algorithm executes the dynamic cone-shaped sweep for each navigation point and generates multiple shortest navigation edges at different directions and ignores other long edges, enabling the improved navigation graph to cover the whole environment area and have less navigation edges. We introduced visibility judgment to avoid that the cone-shaped space cover invalid area (spatial area obstructed by obstacles).

In this study, *NP* and *NE* respectively denote the set of navigation points and navigation edges. *A* is the adjacent matrix representing the topological relations of the navigation points. *O* is the set of spatial obstacles. NP_i represents the i^{th} navigation point in *NP*. *eNP* denotes the set of navigation points excluding NP_i. eNP_j shows the j^{th} navigation point in *eNP*. C_i indicates the cone-shaped space set with NP_i being the central point. $iCone_\mathrm{p}$ represents the p^{th} cone-shaped space in C_i. S_i represents the set of navigation points in C_i the closest to NP_i. The procedure of the algorithm is as follows:

(1) *NP* and *O* is entered to initialize *A*. Mark NP_i and then search *eNP*.

(2) In order to ensure NP_i and eNP_j are visible. We need to check whether the line connecting NP_i and eNP_j intersects with the edges of obstacles in *O*. Parameter equations are introduced in this article [20]:

$$\begin{cases} t=\dfrac{(x_i-x_a)(y_i-y_j)-(x_i-x_j)(y_i-y_a)}{(x_b-x_a)(y_i-y_j)-(x_i-x_j)(y_b-y_a)} \\ S=\dfrac{(x_b-x_a)(y_i-y_a)-(x_i-y_a)(x_b-y_a)}{(x_b-x_a)(y_i-y_j)-(x_i-y_j)(x_b-y_a)} \end{cases} \tag{3}$$

where (x_i,y_i) ,(x_j,y_j), (x_a,y_a) ,(x_b,y_b) are the coordinates of NP_i , eNP_j and the two vertices of one obstacle edge, respectively. When $0<t<1$ and $0<S<1$, NP_i and eNP_j are not visible.

(3) If NP_i and eNP_j are visible, search C_i and execute dynamic cone-shaped sweep operations to process the navigation points. Flow chart of the navigation point processing algorithm based on DCS operations is shown in Figure 5.

Copyright © 2017 SERSC

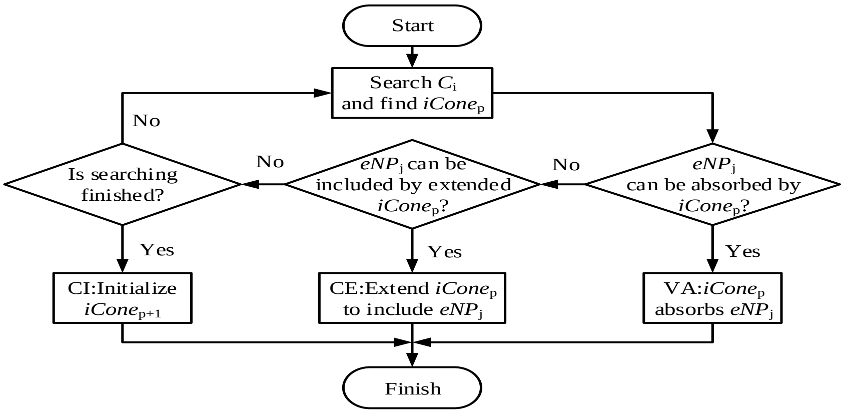

Start
Search C_i and find $iCone_p$
No
Is searching finished?
No
eNP_j can be included by extended $iCone_p$?
No
eNP_j can be absorbed by $iCone_p$?
Yes
Yes
Yes
CI:Initialize $iCone_{p+1}$
CE:Extend $iCone_p$ to include eNP_j
VA:$iCone_p$ absorbs eNP_j
Finish

Figure 5. Flow Chart of the Navigation Point Processing Algorithim Based on DCS Operations

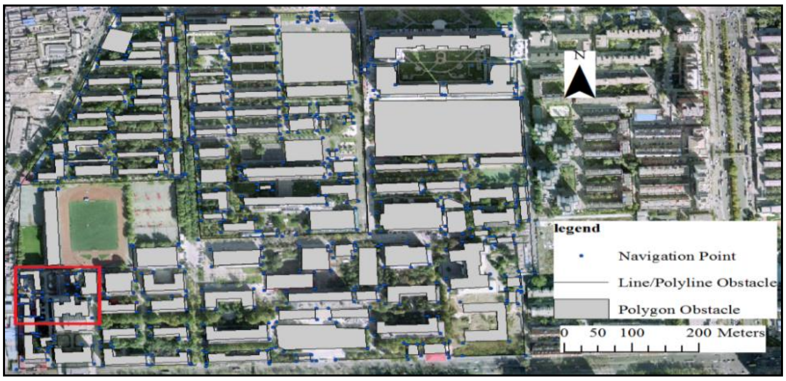
(4) Repeat the step (2) and (3). After the *eNP* searching is over, traverse S_i to extract navigation points the closest to NP_i from C_i to adjust *A*.

(5) Until all the navigation points in *NP* are marked, traverse *A* to generate *NE*, and then build the improved navigation graph *ING* (*NP*, *NE*).

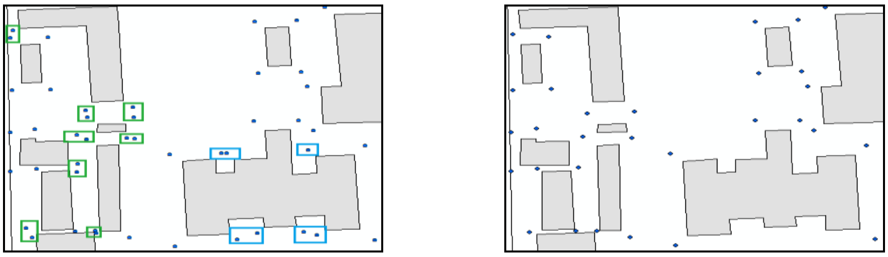
5. Experiment Results and Discussions

We selected the campus of Beijing Forestry University as the study area. Within this area, there are 12 linear obstacles such as the campus walls and rails, and 129 polygonal obstacles, consisting of buildings and green belts. C# language was used to realize the proposed approach based on ArcGIS Engine. As specified above, the navigation points and navigation edges are major factors influencing the performance of the improved navigation graph. Experiments were made on them respectively to analyze the results and verify the effect of the proposed methods.

Figure 6(a) shows all navigation points generated by the proposed approach in the study area. Experiment parameters were: *h*=4m, *P*=5m. Through comparison with the complete navigation graph, the effect of the proposed algorithms is verified. Figure 6(b) and 6(c) respectively show the generated results of the red box area in Figure 6(a) in the complete navigation graph and the improved navigation graph. Comparison result shows that this algorithm is able to avoid generating navigation points at the global concave vertices of the polygon obstacles, as is shown in the blue box in Figure 6(b). In addition, adjacent navigation points can be merged into a single point, as is shown in the green box in Figure 6(b). In this experiment, 585 navigation points were generated, about 11.76% fewer than the 663 navigation points of the complete navigation graph.



(a) All navigation Points of the ING (*h*=4m, *P*=5m)



(b) Part of Navigation Points in the CNG

(c) Part of Navigation Points in the ING(h=4m, P=5m)

Figure 6. Comparison of the Generated Navigation Points

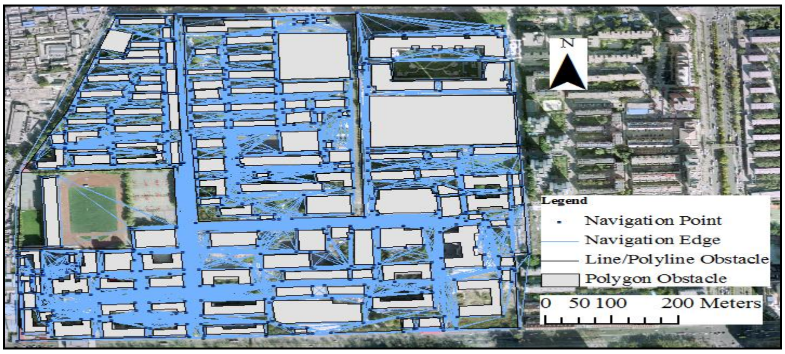
Table 1 shows the number of generated navigation points when the threshold value *P* was set to 5m, 10m, 15m, 20m, 25m and 30m. It can be seen that the number of navigation points in the improved navigation graph is apparently smaller than the complete navigation graph. The higher value of *P* will lead to the fewer navigation points. However, as the mergence operation updates the positions of the navigation points, the merged navigation points' positions have dramatic changes and are quite far away from the corners of the obstacles, not conforming to the pedestrian's wayfinding habits. In addition, when $P\geq25$m, the algorithm mistakenly merged a few navigation points on different sides of the linear obstacles. Therefore, we should set *P* to a reasonable value and maintain a limited number of navigation points at the same time. In this study, we set *P*=15m, and then approximately 27.15% of the number of navigation points can be reduced.

Table 1. Comparison Results of the Number of Navigation Points

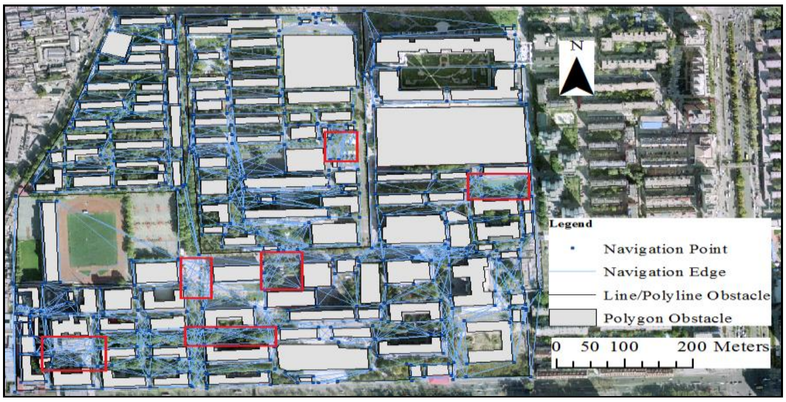
P	5m	10m	15m	20m	25m	30m
N	585	520	483	448	421	399

P is the threshold of the mergence distance, *N* is the number of navigation points.

Figure 7(a) shows the complete navigation graph generated in the study area. The number of navigation edges is 7063. As all adjacency relations among the navigation points are contained, the open space is distributed with dense and redundant navigation edges. Figure 7(b) shows the generated improved navigation graph when the sweep angle threshold *D* was set to $\frac{\pi}{12}$. The number of navigation edges is 1404. It can be seen that this approach is able to significantly reduce the number of navigation edges. The navigation edges distributed around the obstacles can provide pedestrians with avoiding obstacle route guidance. As shown in the red box in Figure 7(b), the intersecting navigation edges in the open area can guide pedestrians pass through the open area from different orientations.



(a) Complete Navigation Graph



(b) Improved Navigation Graph (h=4m, P=15m, $D=\frac{\pi}{12}$)

Figure 7. Comparison of Generated CNG and ING in the Study Area

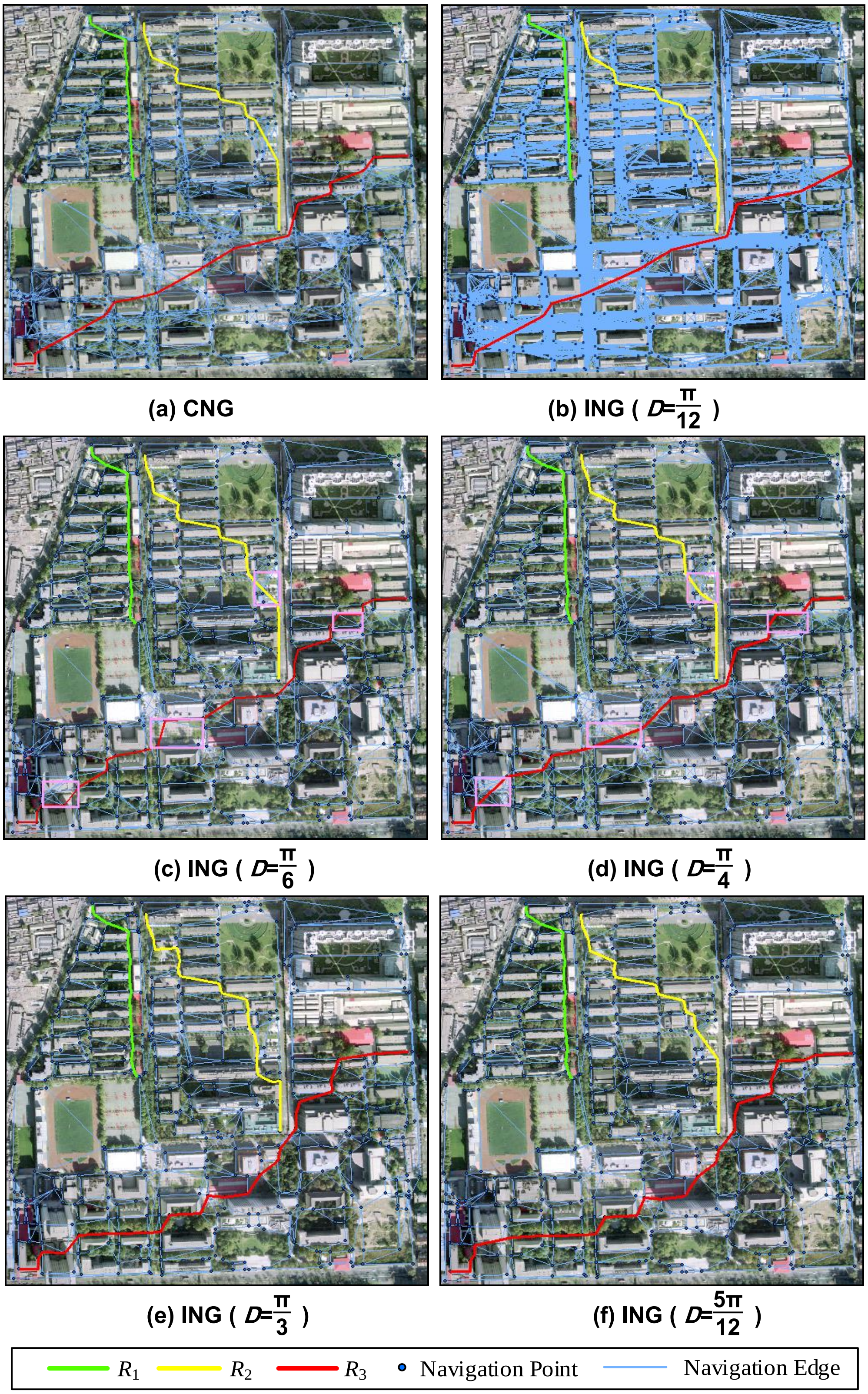
Table 2 shows the number of navigation edges generated using the algorithm proposed in this study when P=15m and D was set to $\frac{\pi}{12}$,$\frac{\pi}{6}$,$\frac{\pi}{4}$,$\frac{\pi}{3}$,$\frac{5\pi}{12}$ and $\frac{\pi}{2}$. It can be seen that this algorithm is able to control the density of the navigation edges by adjusting the sweep angle threshold value. The higher value of D will lead to the less dense of the navigation edges. When D was set to be the smallest value in this study ($D=\frac{\pi}{12}$), the number of navigation edges could be 80.12% less than the complete navigation graph.

Table 2. Comparison Results of the Number of Navigation Edges

D	$\frac{\pi}{12}$	$\frac{\pi}{6}$	$\frac{\pi}{4}$	$\frac{\pi}{3}$	$\frac{5\pi}{12}$	$\frac{\pi}{2}$
E	1404	1125	965	867	850	826

D is the sweep angle threshold value, E is the number of navigation edges.

As mentioned above, the improved navigation graph should provide as accurate as possible route guidance. Figure 8 shows the shortest routes of three OD pairs computed by the complete navigation graph and the improved navigation graph with different sweep angle threshold values. As the dynamic cone-shaped sweep only records the navigation point in the cone-shaped space the closest to the central vertex, R_1 in the improved navigation graphs all consist of short route segments, showing minor twists, but the holistic direction still remain the same. This phenomenon indicates that the routes in narrow area are less affected by dynamic cone-shaped sweep operations. R_2, R_3 in Figure 8(a) and (b) have highly similar geometric shapes. It illustrates that ING($D=\frac{\pi}{12}$) is able to provide the pedestrians with accurate route guidance. As is shown in Figure 8(c) and (d), R_2 and R_3 in the improved navigation graph both have changes comparing with these in Figure 8(a). The deformations of the route all appear in open area (pink box in the figures). This is because higher value of D makes the cone-shaped space absorb some valid navigation points, leading to many unnecessary route nodes in the shortest route. Although these routes cannot guide pedestrians pass through the open area, holistic shapes approximately keep consistent with the routes computed by CNG. When the value of D is too large, the cone-shaped space will include lots of valid navigation points, leading to distinct changes of the geometric shape of the routes. For instance, R_3 in Figure 8(e) and R_2, R_3 in Figure 8(f) have too long route length and detour unnecessary areas, resulting in wrong route guidance for the pedestrians.



(a) CNG

(b) ING ($D=\frac{\pi}{12}$)

(c) ING ($D=\frac{\pi}{6}$)

(d) ING ($D=\frac{\pi}{4}$)

(e) ING ($D=\frac{\pi}{3}$)

(f) ING ($D=\frac{5\pi}{12}$)

Figure 8. Comparison of Three Routes Computed by the CNG and INGs

In a word, the higher value of parameter D will lead to the smaller data size of the improved navigation graph, resulting in a greater impact on the geometric shape of the shortest route of pedestrian. When D is set to $\frac{\pi}{12}$, the proposed approach can generated the improved navigation graph which data size can be largely reduced. The pedestrian network represented by the improved navigation graph can provide the pedestrians with accurate route guidance. Increasing of the value of D will further reduce the data size of the pedestrian network, but resulting in greater impact on the effect of route guidance. Therefore, the value of sweep angle threshold D should balance the demand of computing environment and application scenarios.

6. Conclusions

Pedestrian network is a discrete model of continuous walking environment, which is used to compute the optimal route and provide route guidance. In order to automatically generate the pedestrian network which could meet the needs of users, we proposed an approach for generating pedestrian network based on improved navigation graph. We regard walking environment as space with obstacles. The shapes of obstacles are sufficiently taken into consideration to avoid the generation of invalid navigation points. Adjacent navigation points can be merged through the proposed optimization method. Sparse navigation edges can be generated by the algorithm based on dynamic cone-shaped sweep operations. This study selected the campus of Beijing Forestry University as experiment area. The experiment results indicated that the generated improved navigation graph can largely reduce the data size of pedestrian network and provide pedestrians with accurate route guidance in the process of wayfinding. In conclusion, this approach can effectively solve the network modelling problem in the pedestrian navigation applications.

In addition, this article only simulated for the campus of Beijing Forestry University, and considered the shortest route length as pedestrians' wayfinding strategy. In the future, we will improve this approach to make it is effective for the space with larger size and consider other wayfinding factors of pedestrians, such as travel time, physical consumption and cognitive habits.

Acknowledgments

This research is supported by the Fundamental Research Funds for the Central Universities (TD-2014-02).

References

[1] H. A. Karimi, P. Kasemsuppakorn, "Pedestrian Network Map Generation Approaches and Recommendation", International Journal of Geographical Information Science, vol. 27, no. 5, **(2013)**, pp. 947-962.

[2] C. Gaisbauer, A.U. Frank, "Wayfinding Model for Pedestrian Navigation", 11th AGILE International Conference on Geographic Information Science, University of Girona, Spain, **(2008)**.

[3] P. Kasemsuppakorn, H. A. Karimi, "Personalised Routing for Wheelchair Navigation", Journal of Location Based Services, vol. 3, no. 1, **(2009)**, pp. 24-54.

[4] P. Kasemsuppakorn, H. A. Karimi, "Pedestrian Network Data Collection Through Location-based Social Networks", 5th IEEE International Conference on Collaborative Computing: Networking, Applications and Worksharing, IEEE, Washington, DC, **(2009)** November 11-14.

[5] P. Kasemsuppakorn, H. A. Karimi, "Pedestrian Network Extraction from Fused Aerial Imagery (orthoimages) and Laser Imagery (LiDAR)", Photogrammetric Engineering & Remote Sensing, vol. 79, no. 4, **(2013)**, pp. 369-379.

[6] Y. L. Chen, G. Wu, F. X. Chen, "Route Direction Method of Pedestrian Navigation Based on Landmark", Geography and Geo-Information Science, vol. 31, no. 1, **(2015)**, pp. 17-22.

[7] X. Zhang, Q. Q. Li, Z. X. Fang, "An Approach of Generating Landmark Chain for Pedestrian Navigation Applications", Geomatics and Information Science of Wuhan University, vol. 35, no. 10, **(2010)**, pp. 1240-1244.

[8] Z. X. Fang, Q. Q. Li, X. Zhang, "A Multiobjective Model for Henerating Optimal Landmark Sequences in Pedestrian Navigation Applications", International Journal of Geographical Information Science, vol.

25, no. 5, **(2011)**, pp.785-805.

[9] Z. X. Fang, Q. Q. Li, X. Zhang, S. L. Shaw, "A GIS Data Model for Landmark-based Pedestrian Navigation", International Journal of Geographical Information Science, vol. 26, no. 5, **(2012)**, pp. 817-838.

[10] S. P. Hoogendoorn, P. H. L. Bovy, W. Daamen, "Microscopic Pedestrian Wayfinding and Dynamics Modelling", Pedestrian and Evacuation Dynamics, **(2002)**, pp. 123-154.

[11] Sud, E. Andersen, S. Curtis, M. C. Lin, D. Manocha, "Real-time Path Planning in Dynamic Virtual Environments Using Multiagent Navigation Graphs", IEEE Transactions on Visualization and Computer Graphics, vol. 14, no. 3, **(2008)**, pp. 526-538.

[12] M. Nasir, C. P. Lim, S. Nahavandi, D. Creighton, "Prediction of Pedestrians Routes Within a Built Environment in Normal Conditions", Expert Systems with Applications, vol. 41, no. 10, **(2014)** , pp. 4975-4988.

[13] Kneidl, A. Borrmann, D. Hartmann, "Generation and Use of Sparse Navigation Graphs for Microscopic Pedestrian Simulation Models", Advanced Engineering Informatics, vol. 26, no. 4, **(2012)**, pp. 669-680.

[14] H. M. Choset, "Principles of Robot Motion: Theory", Algorithms, and Implementation, MIT press, Cambridge, **(2005)**.

[15] C. Y. Yang, H. Hu, P. Hu, F. Cao, "Solution of Euclidean Shortest Path Problem Space with Obstacles", Geomatics and Information Science of Wuhan University, vol. 37, no. 12, **(2012)**, pp. 1495-1499.

[16] Y. H .Liu, S. Arimoto, "Path Planning Using a Tangent Graph for Mobile Robots Among Polygonal and Curved Obstacles Communication", The International journal of robotics research, vol. 11, no. 4, **(1992)**, pp. 376-382.

[17] P. K. S. Prakash, D. Ceglarek, M. K. Tiwari, "Constraint-based Simulated Annealing (CBSA) Approach to Solve the Disassembly Scheduling Problem", The International Journal of Advanced Manufacturing Technology, vol. 60, no. 9, **(2012)**, pp. 1125-1137.

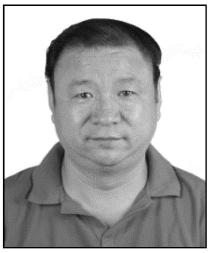
[18] Q. Zhang, J. C. Ma, L. Y. Ma, "Environment Modeling Approach Based on Simplified Visibility Graph", Journal of Northeastern University：natural Science, vol. 34, no. 10, **(2013)**, pp. 1383-1373.

[19] Y. Li, H. Tsukaguchi, "Relationships between Network Topology and Pedestrian Rout Choice Behavior", Journal of the Eastern Asia Society for Transportation Studies, vol. 6, **(2005)**, pp. 241-248.

[20] J. Y. Gong, "GIS Fundamentals", Science Press, Beijing, **(2001)**.

Authors

Yuelu Chen, he is a Ph.D. Candidate of Beijing Forestry University, Beijing, China. His research interests include location-based service, mobile GIS applications and pedestrian navigation system.



Tianzhong Zhao, he is a professor in the School of Information Science & Technology at Beijing Forestry University, Beijing, China. His currently research interests include information resources management, scientific management and decision making, forestry informatization.