A Two-level Path-finding Strategy for Indoor Navigation

Liu Liu and Sisi Zlatanova

Abstract In this paper, a two-level path-finding strategy is presented. It can derive a geometric indoor route according to different preferences. On the first level a room sequence is derived by means of non-metric criteria, such as the number of visited rooms or the number of obstacles in rooms. Based on the sequence, a geometric path with respect to obstacle shape and user size is computed for each traversed room. Then all of these paths compose a final path. The approach is illustrated with an example of a residential building. Compared to other related work, this strategy allows greater flexibility in providing the detailed path within changing indoor environments.

Keywords Indoor navigation · Path-finding · Two-level strategy

1 Introduction

Indoor navigation is a task consisting of three stages. They are path-finding, localization and guiding users. Path-finding is a key step to provide movement information based on knowledge about buildings. Thus path-finding should be conducted on certain model which represents a indoor environment. According to graph-based methods [1–3], certain types of graph can be derived from different

L. Liu (✉) · S. Zlatanova
GIS Technology Section, OTB Research Institute for the Built Environment, Delft University of Technology, Jaffalaan 9, 2628 BX Delft, The Netherlands
e-mail: L.liu-1@tudelft.nl

S. Zlatanova
e-mail: S.Zlatanova@tudelft.nl

S. Zlatanova et al. (eds.), *Intelligent Systems for Crisis Management*, Lecture Notes in Geoinformation and Cartography, DOI: 10.1007/978-3-642-33218-0_3,

building subdivision ways. Most of these graphs belong to Geometric Graph. It means that geometric information (metrics) is attached to nodes/edges of the graphs. Typically, the Combinatorial Data Model (CDM) proposed by Lee [2] is used to derive geometric network by making use of Medial Axis Transformation (MAT) [4]. Then it covers adjacency, connectivity and hierarchical and geometric relationships of a building. A framework named Multilayered Space-Event Model (MLSEM) [5] allows distinct links to be established between different space layers, such like between building geometry and sensor coverage layers. In this way, different representations of a building can be incorporated into the framework.

Another alternative is the Indicative Route Method (IRM) proposed by Karamouzas et al. [6]. It is built on the "corridor map" proposed by Geraerts and Overmars [7]. The "corridor map" structure offers a set of collision-free spaces. IRM can generate routes as smooth skeletons in a "corridor map". Moreover, there is another type of solution that can be named triangulation-based method [8, 9]. Based on a certain triangulation strategy, (indoor) environments are represented by different triangle areas. Thus some sort of a medial axis network can be derived and it would be used for path-finding. Though IRM is a nice method for considering obstacle avoidance, currently the network of indicative route can not be automatically created. Similarly, various triangulation-based methods are also attractive, yet much more artificial nodes may be added in for path-finding compared to graph-based methods.

In our research, we concentrate on the graph-based method for path-finding. However, we intend to avoid constructing the entire geometric graph of a building, because there may be considerable increment of a graph's node/edge number when the graph represents a complex building with plenty of rooms and openings. For instance, a respectable amount of nodes and edges may be added in when the medial axis of a space is computed. Furthermore, the conventional geometric graph is a bit feeble to take indoor obstacles into account. Usually the wide-used geometric network [2] representing the interior of a building lacks a proper representation of obstacles. Additionally, it's not adaptable to dynamically changing scenarios. Even if an indoor environment changes a little, yet a new up-to-date graph needs to be generated. The process would be appreciably time-consuming.

Therefore, it is necessary to develop a strategy to overcome the deficiencies of these widely-used graph-based methods (e.g. path-finding on CDM). A pivotal foundation of our two-level strategy is an appropriate data model representing buildings [10]. The model we used is a logical graph. It only represents the building connectivity. The logical graph (i.e. connectivity graph) could be automatically derived from the building semantics and thus metrics would not be introduced. In our two-level strategy, it will be used in the first level (i.e. the *Rough* level). Then a space/room sequence to be passed would be provided. On the second level (i.e. the *Detailed* level), "door-to-door" paths will be computed for single spaces of the previous sequence.

This paper is organized as follows. In the next section, methods/technical aspects will be introduced. Then the two-level strategy will be presented and exemplified. Finally, we will elaborate on conclusions and future developments.

2 Related Issues

The purpose of indoor path-finding is to obtain different indoor paths according to distinct requirements of various users. A specific path-finding task is labeled by its purpose, such as the shortest path or the fastest path. According to typical graph-based methods (where nodes represent spaces and edges represent openings), a classification of diverse indoor paths is given:

- *Least-effort*: it minimizes the total required travel time for reaching a destination [11].
- *Shortest (shortest-distance)*: it minimizes the required distance between a start and a destination [2, 3].
- *Central point strategy*: finding a path by trying to transit well-known locations (e.g. landmarks) of buildings [12].
- *Direction strategy*: heading to the horizontal position of a destination as directly as possible and regardless of the level-changes [12].
- *Floor strategy*: firstly finding a path to the storey of a destination, then horizontally finding the destination on the storey [12].

In our two-level strategy we apply different criteria for path-finding on the first and the second levels. We will elaborate them in the following subsections.

2.1 Criteria of Path-finding on the First Level

We distinguish between two path types as follows:

- *Least-space-visited*: finding a path by visiting the least number of indoor spaces between a start location and a destination.
- *Least-obstruction*: finding a path guaranteeing the least degree of obstruction between a start location and a destination.

We believe much more other path types could be defined according to specific preferences of various users and the strength of preference can be estimated.

Table 1 lists the merits and drawbacks of indoor pathfinding approaches. As we mentioned before, a *Shortest-distance* path derived from conventional geometric graph usually does not concern obstacles due to the insufficient representation. In this sense the path may not be real closest path between a start and a destination. To get a *Least-effort* path, the type and walking speed of specific users would be reckoned. Yet these estimated data might not be accurate. The *Central point*

Table 1 Comparison of different path types

Path type	Merit	Drawback
Shortest-distance	It is a path providing the minimum distance between a start and a destination	Full building metrics is required to get an accurate shortest path. Sometimes, the shortest path isn't the one that is passable or easy to be followed
Least-effort	It is a path considering the minimum travel time as the 'optimal' criterion	It needs users' type and speed as parameters for computation. Yet usually these parameters are just rough estimations
Central point strategy	It's an easy and practical way for users to find a path by themselves	It may cause considerable detours
Direction strategy	Finding a path by sticking to the direction between the start and the destination	It may result in winding path which can get unfamiliar users lost
Floor strategy	It is a easy-followed pattern to arrive the floor of destination	It may not be the minimum distance path
Least-space-visited	It is a path presenting the least number of visited rooms between assigned start and destination	To follow it may result in more travelling time and distance
Least-obstruction	It is a path considering the least degree of blockage between a start and a destination. Moreover, it has no request for building metrics	Gaining an accurate path of such type needs accurate dynamic information. The information is difficult to be exactly collected in practice

strategy, *Direction strategy* and *Floor strategy* are proposed by Hölscher et al. [12]. They mainly act as strategies to help users find paths by themselves.

Liu and Zlatanova [10] classify three types of indoor obstacle. They are moveable, fixed and dynamic obstacles. The *Least-obstruction* path concerns all these obstacles and it's the potentially least-interrupted route. The *Least-space-visited* path may avoid detours by controlling the number of passed spaces. In this paper, we initially decide to compute the *Least-obstruction* and/or the *Least-space-visited* path on the first level.

2.2 Visibility Graph Construction on the Second Level

In our research we select Visibility Graph (VG) method for path-finding with respect to indoor obstacles [13]. The nodes of a VG correspond to geometric vertices of all polygonal obstacles on 2D plane. A pair of mutually visible nodes is corresponding to a visible edge. Then all of nodes and visible edges compose a

VG. When a Euclidean shortest path is to be computed on 2D plane, the basic approach is to build a VG and apply certain shortest-path search algorithm (e.g. Dijkstra algorithm, [14]) on the graph. Specifically, the weight of each visible edge is the distance of related two nodes.

Alt and Welzl [15] and Overmars and Welzl [16] give an algorithm to support visibility graph construction for polygons in $O(n^2)$ time. This algorithm is better than traditional ones with $O(n^2 \log n)$ time complexity [13]. So it's introduced in this paper. Figure 1 illustrates a VG in a 2D single space/room. The nodes of the VG involve both geometric vertices of obstacles and concave vertices of the space/room polygon.

2.3 Constraint of User Size on the Second Level

A user's size and shape may be critical when a path is computed in a room with many obstacles. In robot motion planning if a robot is represented by a polygon on a 2D enviroment, the real space for its movement is named Work Space (WS) [13]. Reference point is a notion to depict a robot's location. The robot's state can be reflected by the reference point and translation and rotation parameters. A space related to the two parameters is named Configuration Space (CS). Apparently a robot in a location of WS (i.e. real world) may intersect certain obstalce area. Thus the non-obstacle part is named Free Space. In WS the obstacle-free movement of a robot can be represented by certain curve in Free CS. In this case, the Minkowski Sums are used to derive obstacle-free paths in robot motion planning [13]. Moreover, Yuan and Schneider [17] presents another approach. They claim it can test the physical traversability of paths for arbitrarily-shaped users in a 3D environment.

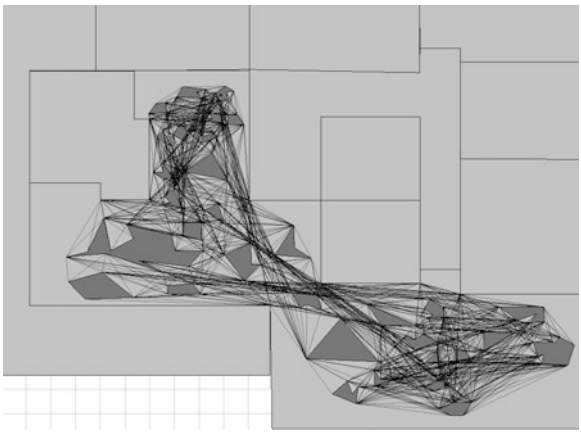

Fig. 1 A visibility graph in a single room

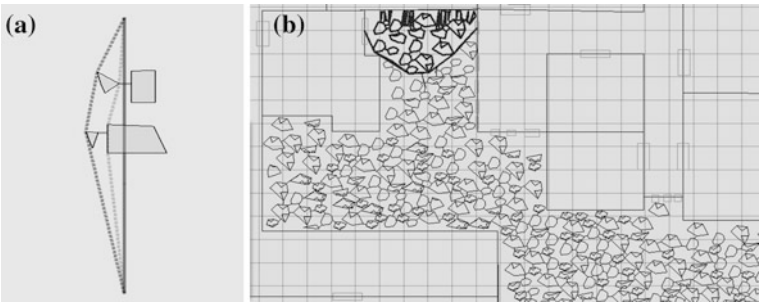

Fig. 2 **a** Paths with respect to user sizes; **b** a path complying with a constraint of user size in a single room

As people are more flexible than robots, it's sufficient to only concern the size constraint of users to report "bottlenecks". These "bottlenecks" involve the minimum gap between any two obstacles. On a 2D plane, the size constraint can be specified as the minimum width of a user. There are different size constraints for diverse users. For instance, the solid line in Fig. 2a is the path without regarding of obstacles, and the middle dash line is the path to avoid obstacles for a child. In Fig. 2a the leftmost dash line indicates the path for an adult. In order to compare user's widths to the "bottlenecks", we detect the minimum distance between any two obstacles. Rotation Calipers algorithm [18] is selected to calculate the minimum distances. Yet it only can be applied to convex polygons. There is an assumption that people would like to move around an obstacle. It means the shortest way for moving around an obstacle is its convex hull. Thus we concern convex hull of obstacles and Rotation Calipers algorithm can be applied.

In Fig. 2b, obstacles in a room and their convex hulls are presented. Minimum distances between all obstacle pairs are computed. If a minimum distance is less than a given tolerance then the related two obstacles are in a same group. As shown in Fig. 2b, any two obstacles in a same group are connected by a segment. When a visible edge intersects any one of these segments, it will be removed. Thus the current user can not pass through the interior of a group of obstacles. The bold line in Fig. 2b is the shortest-distance path between two doors in the room. It rounds a group of obstacles (the highlighted part).

To sum up, exploring different types of indoor path could help us design and test appropriate path-finding results; constructing the visibility graph can assist path-finding which involves indoor obstacles. Moreover, user size is concerned and then available paths can be provided to the user. Based on related aspects mentioned above, in the next section we will elaborate the two-level path-finding strategy and show an example of path-finding.

3 Two-Level Path-finding Strategy

As mentioned previously, the two-level path-finding strategy is a graph-based method. It uses two kinds of graph for path computation. On the *rough* level the logical graph of a building is required. Nodes of the logical graph represent spaces/rooms, while its edges denote openings in the building. On the *detailed* level, a "door-to-door" path in a single space is computed. Thus the nodes represent all openings and transition locations among obstacles in the space, and the edges imply the Euclidean distances.

Compared to the work of Liu and Zlatanova [10], we improve the two-level strategy mainly in following aspects:

1. Metrics are not introduced to the *rough* level path-finding (routing);
2. Obstacles are taken into account in the *detailed* level routing. Related path-finding results are still "door-to-door" paths.
3. The size and shape of users are considered.

As connectivity is the only required information on the *rough* level, it would facilitate an automatic derivation of the logical graph from building semantics. In this case, distances between nodes are not available. Routing result of *rough* level is a space sequence. It implies the spaces to be passed through.

Point 2 indicates the resulting path is the shortest way between two doors of a space. The door-to-door paths are geometric routes in single spaces. According to the space sequence, door-to-door paths in those spaces compose the final path. Therefore, geometric paths are computed on the fly. Point 3 guarantees users can follow the derived paths. In the following subsections, we will illustrate how to apply the two-level strategy for path-finding.

3.1 Rough Level

At first, the connectivity graph of a building should be given. Floor plans of a building with four stories are shown in Fig. 3. Some spaces of the building contain obstacles. There are two elevators and one stairwell as vertical movement spaces.

With the logical graph, paths can be computed according to different attributes of edges. It is also common to compute the value of edges by some weighted function [19]. In order to facilitate the explanation of our method, in this paper we just consider the attribute named *obstruction degree* of nodes. More specifically, the logical graph can be a directed graph to denote available passing directions. If a space (i.e. a node) leads to another space, then its *obstruction degree* value is the value of related directed edge. Finally, the *Least-obstruction* path could be acquired as a space sequence from the first level routing.

Figure 4 expresses the connectivity graph of the building. As shown in Fig. 4, the central part includes the vertical movement spaces. And the centrifugal clusters

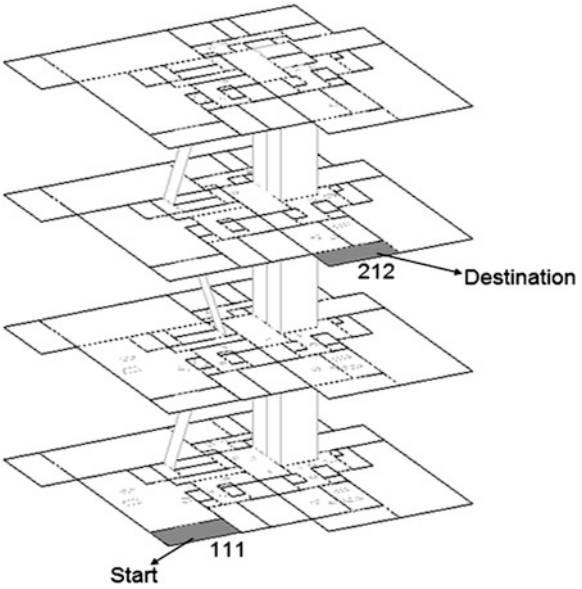

Fig. 3 Floor plans of a test building

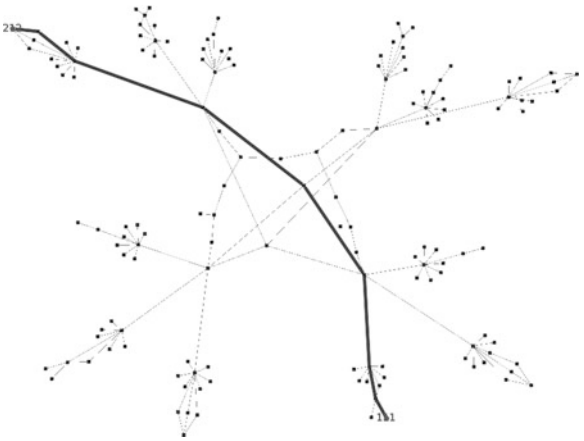

Fig. 4 The connectivity graph of the test building

denote horizontal spaces in different stories. The bold line denotes the *Least-obstruction* path from the node 111 to the node 212. It reveals that an elevator is selected for a vertical transition. It is the conformable and maybe prompt way for users when there is no emergency. Besides, we could eliminate some apparently useless nodes. For instance, if a space is only connected to one other space and it's not the start or destination, then it can be removed. In this way the connectivity graph in Fig. 4 can be simplified, which is shown in Fig. 5.

It is apparent that the number of nodes and edges of a logical graph is almost constant. Yet more nodes and edges would be removed (or added) on the graph

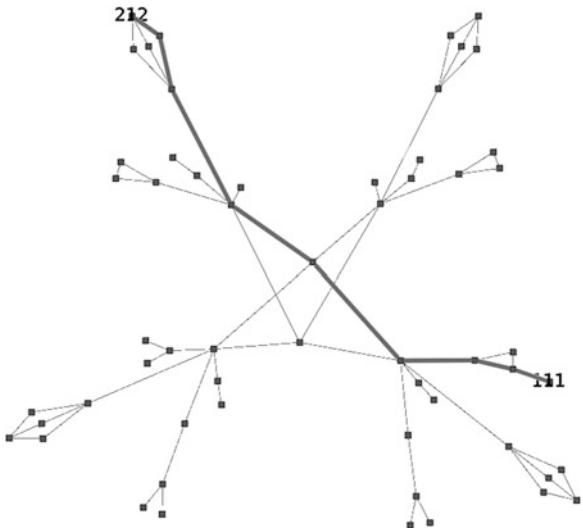

Fig. 5 The connectivity graph after selection

according to dynamic changes in a building. The updating process can be finished on one logical graph, so there is no need to derive a new graph of the building every time. The path derived from the first level is just an indication of spaces to be passed. Yet the movement in the interior of spaces is not clear. In the following subsection, we will introduce the detailed path.

3.2 Detailed Level

After the space sequence is determined at the first level, a detailed path would be searched orderly in each space. Generally, if a space has multiple openings (which mostly are doors) to the next space, then we should compute the door-to-door route for each opening to select the shortest one. For any two openings in a single space, the concrete computation steps are described as follows.

1. Use Rotating Calipers algorithm to compute the minimum distance between any two of obstacles.
2. Construct a visibility graph in the space.
3. Use a given tolerance to filter visible edges. We should have detected the obstacles whose minimum distance is less than the tolerance before that.
4. If the start and the end nodes are invisible, we check whether there are visible edges connecting the start/end nodes. If it's not, there is no path from the start node to the end node.
5. If the path exists, compute the *shortest-distance* path on the visibility graph. Otherwise, return to the previous space and re-compute the space sequence on the first level.

If there is no passable path in the current space, iterative computation of the space sequence starts from the previous space. For each passable space, its detailed path is the shortest path in it. The iteration computation stops when the detailed path in destination space is obtained.

According to different scenarios, several situations in a single space could be categorized:

- No obstacles. If there is only one door to the next space, we just compute the door-to-door path with the shortest distance.
- With obstacles. A visibility graph will be constructed at first. Also, impassable visible edges are removed. Then the *shortest-distance* path to the next space is computed. If there is no path to the next space, then the space sequence is re-computed from the last space.
- With emergency. There may be some unexpected dynamic or new static (like collapsed cupboards) obstacles in current space. The processing is similar to the scenarios with obstacles: if the dynamic obstacles block all possible openings, users should return the previous space and the space sequence is re-determined.

All the detailed paths in these spaces on the space sequence will be put together. This is the final path. Except for the start and the destination spaces, any other space on the sequence has a determined start location as the end location in the path of the previous space. As shown in Fig. 6, the bold line is the final path between the start and the destination spaces. From the top view of its horizontal parts, obviously the path is a door-to-door and obstacle-avoided path.

In brief, the two-level path-finding strategy is feasible and flexible. Compared to the widely-used geometric graph method, it has preferable ability to take indoor obstacles into account.

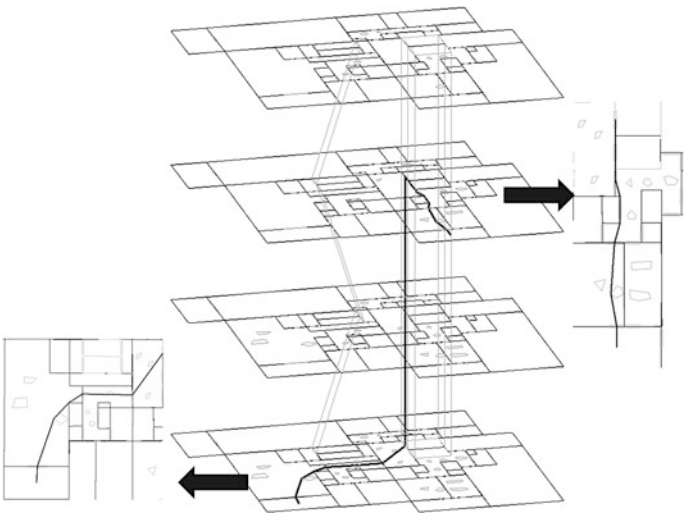

Fig. 6 The detailed path from the start to the destination in the building

4 Conclusions

This paper presents our two-level indoor path-finding strategy. In general, the two-level routing strategy has several primary merits:

1. Path-finding on the first level can be conducted on connectivity graph without metrics and with the limited number of nodes and edges. The number is determined by the structural subdivision of buildings;
2. The detailed geometric network per room is computed only to define the path in the room. Thus extra nodes/edges will not be added to the connectivity graph derived at the first level. Also, the method can support path-finding in dynamic environments as detailed paths are derived on the fly.
3. Obstacles and the shape and size of users can be considered in this method, which is seldom taken into account by other related researches.

When the two-level path-finding strategy is used with a data model of buildings, extra subdivision on buildings would not be introduced (as with the triangulation or MAT methods). That means it's unnecessary to adjust the size of building spaces. Even a spacious room would not be subdivided if it's a basic space in a building. Therefore, the strategy has potentially a broad range of applications.

In this paper, we only show an initial test about single-attribute path on the rough level. It concerns merely one preference. In the future, more criteria of path-finding would be used together to get a space sequence. In this case, weighted functions can be devised with preference attributes, and then the weight value of each edge of a logical graph will be computed with respect to every attribute. The anticipated space sequence is a weighted result as well. On the detailed level, currently only visibility is taken into account for computing a geometric path. Yet it's a purely geometric criterion. In order to concern diverse capabilities of the navigation users (e.g. vision-impaired or wheelchair users), some other factors may be considered to imply traversability in the future.

Currently all indoor spaces are inputted as 2D polygons. In the next step we will investigate how to apply the two-level strategy to 3D solid indoor spaces. Furthermore, its application in emergency scenarios will be explored. On the rough level, connectivity is impacted by emergencies, and then we need an updating mechanism to maintain nodes and edges of the connectivity graph. On the detailed level, the number of dynamic and static obstacles will sharply increase in a short time. Then temporal factors could be introduced to computing detailed paths.

References

1. G. Franz, H. Mallot, J. Wiener, Graph-based models of space in architecture and cognitive science—a comparative analysis, in *Proceedings of the 17th International Conference on Systems Research, Informatics and Cybernetics*, Windsor, Canada, 1–7 Aug 2005, pp. 30–38

2. J. Lee, A spatial access-oriented implementation of a 3-D GIS topological data model for urban entities. Geoinformatica **8**(3), 237–264 (2004)
3. M. Meijers, S. Zlatanova, N. Preifer, 3D geoinformation indoors: structuring for evacuation, in *Proceedings of Next generation 3D City Models* , Bonn, 21–22 June 2005
4. H. Blum, A transformation for extracting new descriptors of shape, in *Proceedings of the Symposium on Models for the Perception of Speech and Visual Form*, ed. by W.W. Dunn (MIT Press, Cambridge, 1967), pp. 362–380
5. C. Nagel, T. Becker, R. Kaden, K. Li, J. Lee, T.H. Kolbe, *Requirements and Space-Event Modeling for Indoor Navigation* (Open Geospatial Consortium Discussion Paper, Open Geospatial Consortium, 2010)
6. I. Karamouzas, R. Geraerts, M. Overmars, Indicative routes for path planning and crowd simulation, in *Proceedings of International Conference on the Foundation of Digital Games*, Orland (2009), pp. 113–120
7. R. Geraerts, M. Overmars, The corridor map method: real-time high-quality path planning, in *Proceedings of the IEEE International Conference on Robotics and Automation* (2007), pp. 1023–1028
8. M. Kallmann, Path planning in triangulations, in *Proceedings of the IJCAI Workshop on Reasoning, Representation, and Learning in Computer Games* (2005), pp. 49–54
9. W.V. Toll, A.F. Cook IV, R. Geraerts , Navigation meshes for realistic multi-layered environments, In *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems* (San Francisco, 2011), pp. 3526–3532
10. L. Liu, S. Zlatanova, A door-to-door pathfinding approach for indoor navigation, in *Proceedings of GeoInformation for Disaster Management Conference 2011*, Antalya, Turkey, 3–8 May 2011
11. J.-C. Thill, T.H.D. Dao, Y. Zhou, Traveling in the three-dimensional city: applications in route planning, accessibility assessment, location analysis and beyond. J. Transp. Geogr. **19**, 405–421 (2011)
12. C. Hölscher, T. Meilinger, G. Vrachliotis, M. Brösamle, M. Knauff, Up the down staircase: wayfinding strategies in multi-level buildings. J. Environ. Psychol. **26**, 284–299 (2006)
13. M.D. Berg, M.V. Kreveld, M. Overmars, *Computational Geometry: Algorithms and Applications* (Springer-Verlag, New York, 2000)
14. E.W. Dijkstra, A note on two problems in connexion with graphs. Numer. Math. **1**, 269–271 (1959)
15. H. Alt, E. Welzl, Visibility graphs and obstacle-avoiding shortest paths. Z. Oper. Res. **32**, 145–164 (1988)
16. M.H. Overmars, E. Welzl, New methods for computing visibility graphs, in *Proceedings of the 4th ACM Symposium on Computational Geometry* (1988), pp. 164–171
17. W. Yuan, M. Schneider, 3D Indoor route planning for arbitrary-shape objects, in *Database Systems for Advanced Applications,* ed. by J. Xu, G. Yu, S. Zhou, R. Unland. Lecture notes in computer science, vol. 6637 (Springer, Berlin/Heidelberg, 2011), pp. 120–131
18. G. Toussaint, Solving geometric problems with the rotating calipers, in *Proceedings of IEEE MELECON'83* (Athens, 1983), p. A10.02/1–4
19. F. Lyardet, D.W. Szeto, E. Aitenbichler, Context-aware indoor navigation, in *Proceedings of the European Conference on Ambient Intelligence* (2008), pp. 290–307