Indoor Semantic Modelling for Routing

The Two-Level Routing Approach for Indoor Navigation

Liu Liu
Indoor Semantic Modelling for Routing

The Two-Level Routing Approach for Indoor Navigation

Liu Liu
*Delft University of Technology, Faculty of Architecture and the Built Environment*
Design: Sirene Ontwerpers, Rotterdam
Cover image: Terminal 2, Beijing International Airport (Liu Liu)

ISBN 978-94-92516-93-0
ISSN 2212-3202

© 2017 Liu Liu (liuliu2679@gmail.com)

All rights reserved. No part of the material protected by this copyright notice may be reproduced or utilized in any form or by any means, electronic or mechanical, including photocopying, recording or by any information storage and retrieval system, without written permission from the author.
Indoor Semantic Modelling for Routing

The Two-Level Routing Approach for Indoor Navigation

Proefschrift

Proefschrift

ter verkrijging van de graad van doctor
aan de Technische Universiteit Delft,
op gezag van de Rector Magnificus prof. ir. K. C. A. M. Luyben,
voorzitter van het College voor Promoties,
in het openbaar te verdedigen op dinsdag 5 december 2017 om 12:30 uur

door

Liu LIU

Master of Science in Cartology and Geographic Information System, Beijing Normal University, China

geboren te Kaiyang, Guizhou province, China
This dissertation has been approved by the
promotor: prof.dr.ir. P.J.M. van Oosterom
copromotor: dr.dipl.-ing. S. Zlatanova

Composition of the doctoral committee:
Rector Magnificus, voorzitter
prof.dr.ir. P.J.M. van Oosterom, promotor
dr.dipl.-ing. S. Zlatanova, copromotor

Independent members
prof.dr. C. Claramunt, Naval Academy Research Institute in France
prof.dr.ir. B. de Vries, Eindhoven University of Technology
prof.dr. P. A. Longley, University College London
prof.dr.ir.arch. I.S. Sevil Sariyildiz, TU Delft, Faculty of Architecture and the Built Environment
prof.dr.ir. P.H.A.J.M. van Gelder, TU Delft, Faculty of Technology, Policy and Management

This work was supported by the China Scholarship Council.
Dedicated to my mother

献给我的母亲
## Contents

Acknowledgments xi

1 Introduction 1

1.1 Motivation 1

1.2 Research objective and scope 5

1.3 Methodology and tools 8

1.4 Outline of the thesis 9

1.5 Overview of related papers to the chapters 11

2 Background 13

2.1 Building models 13

2.2 Navigation models 22

2.3 Routing algorithms and methods 37

2.4 Pedestrian wayfinding behaviours 43

2.5 Indoor positioning and tracking 44

2.6 Adopted methods in this thesis 48

2.7 Summary 49

3 Space modelling for two-level routing 51

3.1 Concept, definitions and terminology 51

3.2 Indoor navigation space model (INSM) 57
# Logical and geometric networks based on INSM

- **3.3** Logical and geometric networks based on INSM
- **3.4** Routing options
  - **3.4.1** Routing using the logical network
  - **3.4.2** Routing using the geometric network
  - **3.4.3** Routing using both networks
- **3.5** Summary

# Routing on logical networks

- **4** Routing on logical networks
  - **4.1** Motivation
  - **4.2** Logical network derivation
  - **4.3** Routing criteria based on INSM semantics
  - **4.4** Routing procedure
    - **4.4.1** Weighted routing
    - **4.4.2** Multi-criteria routing
  - **4.5** Summary

# Routing on geometric networks

- **5** Routing on geometric networks
  - **5.1** Motivation
  - **5.2** Geometric network generation
    - **5.2.1** Approach to consider user size
    - **5.2.2** Edge generation of geometric networks
    - **5.2.3** Compute MD between obstacles and find inaccessible gaps
    - **5.2.4** Group obstacles
    - **5.2.5** Select obstacle groups
    - **5.2.6** Create boundaries for selected groups
    - **5.2.7** Compute the MD between the boundaries of the obstacle group and walls
    - **5.2.8** Create a VG considering inaccessible gaps with walls
    - **5.2.9** Network from edges
    - **5.2.10** Geometric network for users with changing sizes
  - **5.3** Summary

# Realization of one-level routing

- **6** Realization of one-level routing
  - **6.1** Software, data, and user profiles
Acknowledgments

To write the acknowledgements, the last part of my thesis, my mind went through my whole PhD journey. It has been my great pleasure to work with and learn from my incredible colleagues and friends during this journey. There are too many names to thank for their assistance with this thesis, and I would like to express my sincere gratitude to everyone, although I will mention a few of them by name.

First of all, I have the utmost gratitude for my supervisors Peter van Oosterom and Sisi Zlatanova. They have guided me throughout this journey with their vast knowledge and insights into GIS and research. As my promotor, Peter’s patience and support to get me on board are sincerely greatly appreciated. I always enjoyed our discussions about the research, and you enlightened me with valuable advice and inspired me to explore diverse possibilities. The framework of this thesis was confirmed by you and many ideas were sharpened with your help. I am so grateful for everything I learned from you.

As my daily supervisor, Sisi helped me all the time with her outstanding supervision expertise, and she could always draw me back when my faith faded away and focus shifted. We had tons of discussions and wrote many publications together, which formed the essence of this thesis. Meanwhile, being a good friend, I appreciate very much that you had a lot of faith in me to become a better researcher. I thank you for everything.

Many thanks go to all the other colleagues I have worked with. It has been my great privilege and pleasure to work with all of you. I would like to thank Jantien Stoter, the head of the 3DGeoinformation group. Thank you for hosting me in this group and for all your support for my PhD project. Ken Arroyo Ohori, thank you for all your time and help. I really enjoyed all the discussions with you, and your help for preparing this thesis is greatly appreciated. Elfriede Fendel, I sincerely appreciate all your enthusiastic help and support since I started my PhD journey, including the Samenvatting of this thesis. Wilko Quak, it was very pleasant to work in the same office with you. I am grateful for your help in solving my practical problems with your creative thoughts. Thanks to Hugo Ledoux, Filip Biljecki, Zhiyong Wang, Ravi Peters, and Abdoulaye Diakité for the enjoyable time when I was doing this work. Thanks to Martijn Meijers, Edward Verbree, Theo Tijssen, Marian de Vries, Tjeu Lemmens and Radan Suba for all your help. Thanks to the Chinese visiting scholars Hua Liu, Junqiao Zhao, Ying Shen, Yan Zhou, and Yeting Zhang for sharing your insights on this research. Thanks also go to Weilin Xu, Haoxiang Wu, and Kaixuan Zhou for the pleasant cooperation via the Geomatics program.

I had the pleasure of making a research visit to the headquarter of Bentley Systems Inc. at Exton in the US. I would like to sincerely thank Alain Lapierre who initiated and supported this interesting project. Thanks go to Mark Anderson and Mark Smith for hosting me at Exton and providing me with BIM IFC datasets and the mobile development toolsets for this thesis (Chapter 6 & Chapter 7). In particular, I appreciate the co-work with Mark Anderson who enthusiastically directed me throughout the development of the demos (Chapter 7) with Bentley products.
Sincere thanks are paid to the members of my doctoral committee. In particular, Bauke de Vries, Christophe Claramunt, Paul Longley, Sevil Sariyildiz and Pieter van Gelder, thanks for your time to read this thesis and to evaluate this work. It is my honor to invite you to join my committee.

I gratefully acknowledge the China Scholarship Council for funding this PhD project.

I would like to give my thanks to the secretaries/ Secretariat of OTB and urbanism departments for their administrative support during my stay in Delft and for my defense preparations. I also want to thank Franklin van der Hoeven and Véro Crickx for directing me how to format this thesis and for helping me to design its cover.

Sincere gratitude is sent to my close friends in the Netherlands, who have enriched my life throughout this PhD journey. Thank you, Chang Wang, Feijia Yin, Ying Li, Qi Tu, Yue Gao, and Ruijun Deng. I cherish your companion and support, and all the joyful time we have had with meals, travelling and laughter. My thanks also go to my old friends in China: Senyang Deng, Wei Chen, and Wei Shu. I sincerely appreciate your generous support over so many years.

I would like to give the biggest thanks to my beloved family as their support has greatly motivated me to complete this doctorate. Particularly I want to dedicate this book to my mother. Her strong faith in me always helped me through the tough times. No words can describe my love and thanks to her. I cannot thank my father enough for all the unconditional love and for encouraging me to pursue meaningful objectives. Also, I extend my special gratitude to my grandfather, who has lit up my road to knowledge since my childhood and continues to do so. Finally, my heartfelt gratitude goes to my big family for their continuous support: my grandmother, my aunties Ruyu, Ruling, Ruyao, my uncles Rujin, Lunqiang and Ruheng, my cousin Yang and all the other family members.

October, 2017

Liu Liu
1 Introduction

This chapter presents an introduction to this PhD research. In Section 1.1, the motivation of this research is presented by defining the research problem, presenting the scientific gap and introducing a possible solution. Section 1.2 presents the main research question of this research, defines the research objective and delimits the scope of this research. Section 1.3 introduces the methodology and tools adopted for this PhD research. Section 1.4 outlines the structure of the whole thesis, giving a short introduction to all of the coming chapters. At the end of this chapter, Section 1.5 lists the author’s own publications and relates them to these chapters of this thesis.

§ 1.1 Motivation

Nowadays, there is a growing need for indoor navigation in large public buildings. According to the Environmental Protection Agency, 75% of the world’s population lives in cities and nearly 90% of their time is spent indoors [Age09, Har12]. Humans perform many activities indoors related to work, shopping, leisure, dining, sport, etc. The buildings and the large variety of associated spaces such as underground passages, sky bridges, metro lines, garages, and intermediate platforms are becoming conglomerates of enclosed spaces (Figure 1.1). This complexity poses many challenges for building managers, occupants and visitors. Indoor navigation (e.g., finding paths to a certain location) and location-based services are some of the most important services indoors.

FIGURE 1.1 An example of the interior of a building (from: www.core77.com/posts/10070/winners-of-german-concrete-competition-10070)
Indoor navigation is an activity where users (e.g., robots, humans or vehicles) navigate to certain locations inside an indoor environment. Indoor navigation is a broad research field which includes five main topics: 1) indoor positioning and localization; 2) indoor modelling for navigation models; 3) algorithms for indoor path computation; 4) human spatial cognition and wayfinding and 5) indoor guidance instructions (e.g., verbal or graphic directories in an interface). Indoor positioning and localization provide user locations, navigation models represent indoor environments, indoor path-finding or routing is conducted in the navigation models to find the optimal or customized path to the target location, and guidance techniques interpret the computed path as directives that a user can follow. If guidance is not available and the user is a pedestrian, she/he can also orientate and navigate by wayfinding strategies that are based on the user’s cognition of the indoor environment.

Among all these indoor topics, it is essential to acquire an appropriate navigation model representing the geometry, topology, semantics (i.e., meaning of spaces) and other context information for indoor environments, and to provide appropriate routing results for different users. The other aspects of indoor navigation are closely related to the navigation model and/or routing: positioning results (i.e., user locations) are visualized in the navigation model. Guidance directives rely on routing results [RZC14]. The wayfinding process needs semantic information (e.g., signage or turns) from the navigation model. A well-developed indoor navigation model should preserve enough crucial information from an indoor environment.

Indoor navigation models represent the interior of buildings in an abstract way, yet they contain sufficient information for conducting navigation tasks. Two types of models can be distinguished, that is, network (vector) and grid (raster) [ZLS+14]. Network models are more widely-used in pedestrian indoor navigation, while grid models are predominant in robot navigation. Topological relationships, geometric information and semantics have commonly been employed for indoor navigation on network models [Wor11]. However, the details of topology, geometry or semantics represented in the reported network models differ significantly. There are two basic groups of network models: 1) networks that preserve the geometric shapes of buildings [MJ05, LOS06a, MZP05, PZ05, SLO07]. The length of paths can be measured in these networks; and 2) networks concentrating only on the connectivity of buildings [BD05, BS01, FMB00, GSC+05, HD04, LL08, JS02, RWS11, SSO08, YCDN07, HOP+08]. The first group of models are more suitable for visualising paths since the paths include accurate coordinates inside the building. However, the first group of models are not suitable for a complex and large building or building composite (i.e., an aggregation of buildings), because the scale of the models can be too large for presentation on screen and computation in memory. The second group of models results in more compact representations that are very convenient for conceptual analysis. Their scale is small but building geometry is not used, thus they cannot support an accurate geometric description of the paths.

A combination of the above two types is the hybrid model, such as hierarchical graphs [LOS06b, SSO08] which can structure and represent indoor spaces/objects and their relationships in different levels according to spatial granularity (e.g., building, floor, rooms and subrooms). The hierarchical graphs include both the topology and geometry of buildings which are organized in different hierarchical levels. However, research on hierarchical graphs focuses on different hierarchical representations of buildings and only a few related routing methods are discussed. In general, paths can be com-
computed in hierarchical graphs of a building with a shortest-path algorithm. However, for a building with many levels, the routing results in different levels needing to be combined and it might be difficult to handle multiple path choices between two locations in the hierarchical graphs [SS008].

Another important topic is to calculate an appropriate path for a given user with her/his own capabilities (i.e., profile). Currently indoor routing research is concentrating on geometric-related features (distance, time cost, and the fewest turns). However, the shortest-distance path in an indoor environment is not as important as those for outdoor environments. Also, a visitor in a building may walk via a longer route through an inquiry point, and then move to the destination. In many reported studies a complete indoor navigation model is built for the buildings [Lee04, MZP05, JTY11]. These networks are designed to meet the basic requirement that indoor routes can be computed for walking users. One complete navigation network of a building is not sufficient for all users (e.g., walking and movement-impaired) and their different tasks (e.g., crossing a specific space or obstacle-avoidance). Navigation networks need to include the flexibility to adapt to user needs and tasks.

FIGURE 1.2 The complex interior of Schiphol Airport, Netherlands (from: www.wikiwand.com/nl/Luchthaven_Schiphol).

Building semantics can support routing to meet the needs of a user or a group of users. Compared to the outdoor paths, there are fewer options for indoor paths. Indoor paths always involve some connection spaces such as corridors, elevators, and stairs, and the number of these spaces are finite. Indoor routing needs to focus on these prominent indoor spaces reflected by their semantics. In the context of indoor navigation, the semantics of a space refers to its functionality for routing. Though semantic models of buildings such as Industry Foundation Classes (IFC) of BIM (Building Information Modeling) [IAI16], and City Geographic Markup Language (CityGML) LoD4 [GKNH12] already exist, they are not specifically designed for indoor navigation. In these models the semantics of building elements (e.g., rooms, doors, and floor surfaces) are abundant
yet the functionality for indoor routing is less involved. The IndoorGML [LLZ+14], a more recent standard of the Open Geospatial Consortium (OGC), makes a good start to structure the semantics of indoor spaces for indoor navigation. However, in this thesis I look for more specific space semantics, which allows not only the description of indoor spaces and their relationships, but can be used to specify routing criteria for different types of user.

The major challenge with indoor modelling is to represent the complex indoor environment (see Figure 1.2) for pedestrians and conduct routing according to user demands. The complex indoor environment includes irregular shapes, open spaces, ‘sub’ spaces (a store in a large hall), complicated obstacles (e.g., the small steel pillars in front of the escalators in Figure 1.2) and different types of passages (e.g., elevators, stairs, and escalators, large halls, long narrow corridors, and sky bridges). Typical examples are the terminals at airports, e.g., Schiphol Airport. A terminal has an irregular geometry and interior spaces separated by many columns, counters, etc. It seems disorganized to most people unfamiliar with it. In this case, semantics are needed to classify all these spaces and objects and to reflect their functionalities for routing. In addition, indoor routing needs to consider user profile and preference [ICC12]. For example, in the same indoor environment a walking user and a wheelchair user correspond to spaces with different semantics and geometry (Figure 1.3).

Compared to the outdoor environment, an indoor environment of a single building as a composite is smaller in size but the complexity is increased since it represents three-dimensional (3D) buildings. Indoors spaces tend to have many obstacles (furniture, columns, podiums, etc.) that can be avoided in various ways, which increases path choices. Users face a larger number of options to go from locations A to B. In such a case, indoor paths are not only related to distance but also dependent on user preference and possibilities. In addition, as a complex building contains many obstacles, two users with distinct sizes need different obstacle-avoiding paths, such as for a user driving a vehicle in the airport and another traveller with luggage. Figure 1.3 presents two different users with different sizes – a walking person and another one with a wheel-

**FIGURE 1.3** Different accessible spaces for distinct users. The blue volumes are considered independent functional spaces, and the others are free spaces which can be accessed without restrictions. (a) A walking user can go above or crawl under the desk; (b) a wheelchair user needs to avoid the desk.
chair. They need different accessible paths. In addition, the walking person can get up on or under the desk, while the wheelchair user can only avoid the desk.

To solve this problem, it is necessary to avoid constructing the entire navigation network of a building again and again for every specific type of user. There can be considerable increment of nodes/edges of a navigation network for a different user when the complex building has plenty of rooms, openings and objects. It is not necessary to store and maintain a large-scale complete network for routing explicitly for all types of users. Furthermore, a complete navigation network [JM05] is insufficient to take into account changes of indoor obstacles and users. For example, when indoor obstacles are moved, a new accurate navigation network needs to be generated. A new user with a dissimilar size also needs a new navigation network since the accessible area varies in terms of the new size. The re-computation process to create navigation networks can be time-consuming for complex buildings. Even if the computation is fast, the navigation network still occupies unnecessary storage space. Thus, a more flexible navigation model is needed to represent complex buildings.

My solution is to create a navigation model which separates the semantics and geometry of buildings into two levels. Unlike hierarchical graphs, the new model does not create many levels according to space granularity. The semantics of buildings are used to find conceptual paths and they can be readily adjusted according to user preferences and decisions; the geometry is applied to network creation after a conceptual path is defined, and the network is suitable for users with a given size. In this way, only a part of space geometries is employed to construct the network for a whole building. In the next section, I will present my main research question and the research objective, which is related to the navigation model of the two levels.

§ 1.2 Research objective and scope

As shown in the discussion in the previous section, in complex indoor environments human users need diverse indoor routing yet a small number of buildings are equipped with a basic routing method, and thus the research question of the thesis is raised, i.e., what indoor routing approach can provide accessible paths according to human user preferences by using the semantics of indoor spaces, in addition to using building topology and geometry?

The main research question is subdivided into several sub-questions which present the details of this research. To be able to conduct indoor routing for different buildings, the first sub-question is posed:

1. What kind of information, data models and routing algorithms has been used and developed so far, and what are their limitations for large complex buildings? (Chapter 2)

Building data sources (e.g., a 3D digital building model as a Computer Aided Design file) provide abundant information but they are not structured for indoor routing. Building models can provide geometry, topology, semantics and other attributes of buildings which can be stored in indoor navigation models. I need to check the current available data models of building and the corresponding navigation models. In addition, I have
to investigate indoor routing algorithms to clarify their application scopes. In particular, I focus on the use of these models and algorithms for large complex buildings. Possible improvement will be discussed if these models and algorithms are not suitable for complex buildings. Thus I introduce the next sub-question:

2. What data and navigation model is appropriate to represent the semantics, topology and geometry of indoor spaces? (Chapter 3)

I need proper data and a navigation model for routing. In this research I investigate network-based navigation models and aim to develop the one that can be easily derived, stored and updated. The navigation model should contain sufficient building information to support routing, such as the semantics and the geometry of indoor spaces, the other building components (e.g., doors, windows, and walls), and indoor objects (e.g., furniture and pillars). The connectivity among the spaces, building components and objects also needs to be provided in a simple and efficient way because this facilitates routing in the next phase. Then this leads to the next sub-question:

3. What kind of user-related paths can be computed with the semantics, topology and geometry of indoor spaces? (Chapter 3)

Routing on the navigation model should meet user preferences/profiles, such as passing specific spaces/locations, using specific types of spaces in a path and avoiding obstacles to follow an accessible path for a user's size. This raises the next sub-question about the details of the routing approach:

4. What kind of routing criteria can be built (or specified) by using the semantics of indoor spaces? (Chapter 4)

To cater for different users, routing criteria need to be formed by incorporating them with the semantics of indoor spaces. As I aim to use the semantics of indoor spaces directly for routing, I investigate the relationships between building components and objects, and then identify their functionalities for routing. Different users may have distinct preferences on space semantics. Considering user sizes, I propose the sub-question:

5. Which approach should be used to compute the exact geometric description of accessible paths according to the size of a user? (Chapter 5)

In order to obtain accessible paths for a user, a routing algorithm is needed to support avoiding indoor obstacles with regard to the user’s size.

6. How are the new proposed user-related paths implemented and applied to realistic cases? (Chapters 6 & 7)

To obtain indoor paths and verify the use of the proposed user-related paths, tests on the generation, efficiency and shapes of these paths are going to be conducted in realistic indoor environments. Different tests are conducted in both desktop and mobile development environments.

The objective of this research is to develop a flexible indoor user-oriented routing approach based on a new type of indoor navigation model. The navigation model should
reflect the semantics, topology and geometry of buildings, and routing can be used efficiently and flexibly in computing the user-related paths.

Within the scope of this research not all aspects of indoor navigation can be covered. The following topics are considered:

1. Design a data model to structure and store space semantics, topology and geometry of buildings for indoor routing.
2. Design an indoor navigation model for routing execution, representing both the connectivity and geometry of indoor spaces.
3. Design a routing approach that considers user-interested spaces/locations and avoids obstacles according to user sizes.
4. Design and develop applications to demonstrate and assess the use of the new routing approach.

The following topics are related or supportive but not addressed in this thesis:

1. Indoor positioning techniques.
2. Building data validation and repairing. The source data are used as-is.
3. Path planning on polyhedral building models or routing on 3D discrete models (e.g., voxels).
5. Indoor wayfinding.
7. Verbal and textual guidance for pedestrians.
8. Navigation for robot and flying objects (e.g., drones) in buildings.
10. Prediction of indoor environmental changes (e.g., the dissemination of smoke).
11. Evacuation planning and navigation in an emergency response. I provide routing for a user or a group of users in a normal state, instead of for all people in a building in an emergency.
12. Integration of indoor and outdoor routing, such as aligning the indoor coordinate system to outdoor ones.
§ 1.3 Methodology and tools

To be able to answer these research questions, I have used a research methodology organized in a number of phases to conduct this PhD research. The methodology is presented as follows:

(1) **Literature review** (Chapter 2).

In Chapter 2, I first review current indoor navigation models and investigate their pros and cons and applications. I also investigate the relationships between navigation models and building models (e.g., data in the CAD, IFC and CityGML formats). Then generation methods of navigation models are compared with respect to the geometry, semantics and topology of building models. Routing algorithms and methods are reviewed to present existing criteria for indoor routing. Following that, indoor wayfinding research is briefly described to show some strategies for indoor navigation. Finally, I introduce the indoor positioning and tracking techniques currently being reported.

(2) **User requirements are derived on the basis of a literature study** (Chapters 2 and 3).

By comparing different indoor routing methods, I can distinguish the conditions and applicable scenarios of these routing methods. Uncovered user needs can be found and the evidence from this lays the foundation for the design of a new indoor routing approach (Chapters 2 and 3).

(3) **A data model managing building semantics and a new indoor navigation model are designed** (Chapter 3).

To be able to understand and follow a path, a human is inclined to grasp the semantics and relationships among spaces, instead of the geometric details of paths (e.g., distance and turns). I define and organize the necessary semantics of indoor spaces to represent different navigational functionalities for users. These semantics are the essence of the data model. The data model is designed to structure the adopted semantics of indoor spaces and objects and to support the automatic creation of the related navigation model. This navigation model contains the semantics, connectivity and geometry of indoor spaces.

(4) **A routing approach conducted on the navigation model is developed** (Chapters 4 and 5).

Based on the navigation model, I design a routing approach that provides paths in the two levels, in terms of the space preferences of users and user sizes. Considering users’ space preferences, I investigate routing criteria to incorporate the semantics of indoor spaces. I also develop and test a method to compute obstacle-avoiding paths for users with a given size. For different applications, routing options are defined by combining routing in the two levels together.

The routing approach is designed on two levels: an abstract and a detailed level. The abstract level represents the semantics and connectivity of indoor spaces and the detailed level is related to space geometry. Routing on the two levels is combined for a
user’s requests on the functional use (e.g., a path crossing as few stairs as possible) and geometric details (e.g., obstacle-avoidance) of indoor paths.

(5) Implementation (Chapters 6 and 7).

Implementation is applied to realistic routing cases. A prototype is developed to conduct the routing approach for different buildings, where indoor navigation models are created and the two-level routing is conducted. The source building models are converted into the proposed data model, and in the prototype the navigation models are generated automatically from the data.

Besides the desktop version of the routing approach, a test with a mobile application is presented as well. This mobile version is portable and thus more convenient to users, which also shows the feasibility of the proposed routing approach.

(6) Analysis (Chapter 7).

This part assesses the proposed solutions. I present the reflection on the obtained results within this research. Several routing options are realized in the prototype and their results are compared with respect to: the number of paths, path distances, and computational cost. A discussion on the results is given to analyse the suitable applications of the routing approach.

For this research I adopt diverse types of building models, and use different software and development tools, which include:

- **Building data**: Representations of floor plans and digital architecture plans, and 3D building models including IFC [IAI16] and CityGML LOD4 [GKNH12].

- **Programming language**: C++, Python.

- **Integrated Development Environment (IDE)**: Visual Studio v10.0 of Microsoft Inc. [Inc17].

- **Visualization tools**: the software Bentley MicroStation V8i [Sys16a], and a Python library igraph [ict16].

- **Mobile application tools**: Bentley Navigator Mobile, Bentley MicroStation Mobile SDK [Sys16b].

The IFC and CityGML LoD4 data will be introduced in Chapter 2, which contains the topology and semantics and 3D geometry of buildings. C++ and Python are two well-known object-oriented programming languages. For the routing application developed in C++ language, I compile it in the Microsoft IDE Visual Studio v10.0. MicroStation V8i is used to visualize the building models, create valid geometry, run the developed routing application and visualize the routing results. The igraph is a network analysis package which is adopted to visualize the navigation network on an abstract level. As mentioned before, I test my solutions in a mobile application. Bentley Navigator Mobile is a mobile application of Bentley Systems, and Bentley MicroStation Mobile SDK supports me to develop indoor routing functions in Navigator Mobile.
§ 1.4 Outline of the thesis

The outline of the thesis is illustrated in Figure 1.4. Chapter 2 gives the background of this thesis, introduces the related work and elicits the requirements of a new navigation model. In this chapter I identify the differences of the current indoor modelling and routing methods, and present the needs for a new routing method.

Chapter 3 presents a new routing method named two-level routing. The two levels refer to the abstract (logical) and detailed (geometric) levels. Two independent navigation networks on the two levels are introduced for routing considering user demands. A data model is designed to support the two levels: it defines the semantics of indoor spaces according to navigational functionalities, and contains the topology and geom-
etry of indoor spaces. This data model facilitates the efficient creation of the two types of network. Different routing computations on the two networks are also presented and illustrated with examples.

Chapter 4 addresses the routing on the abstract (logical) level that uses the semantics of indoor spaces and the network topology (connectivity of spaces). This chapter presents the different criteria and the computation of routing on the abstract level.

Chapter 5 presents the routing considering a user’s size on the detailed (geometric) level, and focuses on the creation of navigation networks on the detailed level.

Chapter 6 introduces data preparation for implementing the two-level routing approach, presents the generation of individual navigation networks from the adopted building data, and shows separate routing tests on the abstract and the detailed levels (i.e., one-level routing).

Chapter 7 illustrates the applications of the developed tools in this research to data in the real world, which shows the results of the two-level routing. This chapter discusses the tests with various indoor data (both 2D and 3D representations). I compare the results of the two-level routing in different use cases, and discuss the improvement on the implementation of the two-level routing.

Chapter 8 provides some discussions on the whole research and concludes this thesis with some future work.

§ 1.5 Overview of related papers to the chapters

My publications that relate to chapters of this thesis are listed in Table 1.1.


TABLE 1.1 My publications and the related chapters of this thesis.
2 Background

Chapter 1 presented the motivation, research question, and methodology of this PhD research. This chapter gives an overview of the essential components of an indoor navigation system and the work related to this PhD research. They are building models and related navigation models, routing algorithms and approaches and research on behaviour modes of humans for routing as described in literature. In addition, indoor positioning techniques are introduced.

Firstly Section 2.1 presents the representative data models of buildings, and Section 2.2 presents existing indoor navigation models including indoor space subdivision, dual graphs, network-based models and grid-based models. Section 2.3 introduces path computation algorithms and a number of routing approaches regarding navigation models. Then Section 2.4 briefly introduces human wayfinding behaviours which reveals the factors for humans to search for a path. Section 2.5 introduces indoor positioning techniques. Section 2.6 presents the building models, navigation networks, and routing methods that were used for this research. Finally, Section 2.7 summarizes this chapter by responding to the first research sub-question (see Chapter 1). This chapter is related to the following author’s own publications: [ZLS13, MZLC14, ZLS+14, LXPZ15].

§ 2.1 Building models

The internal structure of buildings is always described by geometrical models (such as Computer Aided Design (CAD) models) and 3D GIS data models [Lee01]. For instance, CAD models contain the pure geometry of buildings in 2D or 3D forms (e.g., lines, polygons, and solids). CityGML is an Open Geospatial Consortium (OGC) standard for semantic 3D city models, as a common information model for the representation of 3D urban objects [GKNH12]. CityGML can represent urban terrain and 3D objects in five levels of detail (LOD). CityGML LOD4, which specifies architectural models (interior of buildings), is used for representation of indoor environments (e.g., rooms, stairs and furniture). CityGML LOD4 can provide semantically rich, object-based building models. Kolbe et al. [KGP05] apply CityGML to various disaster management applications and demonstrate how the connectivity among rooms for pedestrian access can be extracted using the shared openings (doors) between rooms.

Another group of digital building models is Building Information Models (BIM) which are developed for covering all the stages of the building lifecycle (from design to maintenance). Industry Foundation Classes (IFC) is an industry standard of BIM [IAI16, NSK09, IZ09], which stores both geometric and semantic information. Based on the abundant 3D geometric and semantic information (thickness, material, direction of opening, etc.) of BIMs, it is possible to automatically derive the required navigation models from
BIMs [DL08]. These two types of models (CityGML and BIM) will be extensively mentioned in this thesis.

Semantics is pivotal for indoor pedestrian navigation. The semantics of spaces reveals their meaning and functions of different building components. For example, a corridor is suitable for a user to transfer from one office to another. In the latest version v2.0.0 of the CityGML LOD4 (see Figure 2.1), classes containing space semantics support indoor navigation include Room, FloorSurface, GroundSurface, Door, BuildingFurniture, IntBuildingInstallation, and InteriorWallSurface [GKNH12] (see Figure 2.1). These classes can be used to derive a navigation network of a building. Classes of Room, FloorSurface and GroundSurface (Figure 2.1) refer to the navigable spaces/surfaces in the building. Classes of Room, InteriorWallSurface and Door (see Figure 2.1) can be used to infer the connections among space (i.e., instances of Room). For example, an InteriorWallSurface instance may contain a Door instance, and the Room instance bounded by this InteriorWallSurface links to the Door instance. In this manner, all the Room instances linked via this Door are connected. Classes BuildingFurniture and IntBuildingInstallation (see Figure 2.1) represent indoor static obstacles since an indoor route needs to avoid them. Different vertical passages (e.g., Stair, Elevator and Escalator) of a building are specified by pre-defined codes for IntBuildingInstallation in CityGML.

The hierarchy of the CityGML LOD 4 semantics are organized in its schema (see Figure 2.2). Firstly, Room elements can be found with the building’s property interiorRoom; secondly, the InteriorWallSurface elements can be found via the boundedBy property of a Room element; thirdly, the opening property of an InteriorWallSurface element contains one or more Opening elements (i.e., Door and Window). In addition, the other two properties interiorFurniture and roomInstallation of the element Room indicate the related BuildingFurniture and IntBuildingInstallation elements of the room, respectively.

CityGML is a uniform data model for city objects including buildings, and it is the ideal model for visualizing computed paths [VDMF09]. CityGML provides geometric and semantic information for indoor navigation, and specific classes which can be used for both navigational and visualization purposes. But it lacks navigation network (graph) models for the stored building data (both geometry and semantics).
FIGURE 2.1 UML diagram of the building model of CityGML. Names of elements without prefix are defined within the CityGML Building module (LOD4) (from [GKNH12]). The red rectangle refers to classes regarding navigable space and surfaces. The green dashed rectangle indicates a connection among spaces (Room, Door, and InteriorWallSurface). The blue dashed rectangle refers to static obstacles (BuildingFurniture and IntBuildingInstallation). The yellow rectangle represents vertical passages.
FIGURE 2.2 The embedded structure with indents of a CityGML document (from [LZ13a]).

FIGURE 2.3 Subset of IFC classes for topographic spaces (from [BNZK13]).
IFC contains abundant semantics of buildings which can be utilized for indoor navigation (see Figure 2.3). The IFC includes several hundred entity classes in an entity-relationship model [IAI16, Wik16], but only a small part of classes are directly relevant for indoor navigation. Taking the version of IFC2x Edition 3, these essential entity classes include IfcSpace, IfcWindow, IfcDoor, IfcStair, IfcTransportElement, IfcFurnishingElement, corresponding to indoor space, window, door and stair, elevator/escalator and furniture/objects, respectively. In particular, IfcTransportElement covers elevators, escalators, moving walkways, etc. Moreover, classes such as IfcRelSpaceBoundary and IfcRelContainedInSpatialStructure describing relationships are important for indoor navigation. As IfcRelSpaceBoundary describes the bounded relation between IfcDoor/IfcWindow and IfcSpace, thus the connections between doors and indoor spaces can be derived from the IfcRelSpaceBoundary classes. In addition, IfcRelContainedInSpatialStructure provides a relationship that an indoor object is contained in an indoor space or a building floor, i.e., an IfcSpace instance containing multiple IfcFurnishingElement instances which can be obstacles to pedestrians. However, the IFC model lacks content of indoor navigation networks, as well as path planning information. Network primitives (nodes and edges) need to be extracted from instances of IFC classes, when the IFC model is applied for indoor navigation.

The semantics of the above two data models have a lot of overlap but they are not completely the same. They provide the 3D building information models in a comprehensive view of the geometric, cartographic and semantic aspects. Commonly many terms in the two different standards point to the same or similar semantics. For instance, there may be several different names for the same type of space in functionality (e.g., corridors, passages and entrance halls). But the IFC and the CityGML LoD4, which are the data models for buildings, do not include indoor navigation networks. For indoor navigation, it is necessary to develop a specific data model unifying space semantics which correlate to navigation networks, which can facilitate the generation of navigation networks from building models.

FIGURE 2.4 Indoor Navigation model (from [BNZK13]).

Brown et al. [BNZK13] provide a concise description on indoor spaces according to their navigational functionalities (see Figure 2.4). The model clearly distinguishes be-
tween obstacles and spaces for navigation (e.g., transition and indoor spaces). Lee et al. [LLZ+14] present a generalized data model named IndoorGML regarding indoor spatial information (which is now already an OGC standard), specifically for navigation purposes. The IndoorGML data model includes two parts: the Core module including basic concepts on space, and the Indoor Navigation module (see Figure 2.5) which focuses on the semantics of spaces for indoor navigation. The class NavigableSpace of IndoorGML denotes generic navigable spaces. One of its subclasses GeneralSpace refers to common independent rooms, and another subclass TransferSpace has three other subclasses: ConnectionSpace, AnchorSpace, and TransitionSpace. ConnectionSpace mainly refers to the thick doors regarded as 3D spaces. Specifically, AnchorSpace depicts the connections between indoor and outdoor worlds, such as an entrance door of a building. Either a stair or corridor, or even a part of them, can be classified as TransitionSpace. The Indoor Navigation module also provides classes to specify paths, such as Route. I have also developed another semantic data model indoor navigation space model (INSM) with more specific functionalities of spaces for indoor routing [LZ12], which will be elaborated in Chapter 3.

**FIGURE 2.5** The UML diagram of the Navigation module (in green) in IndoorGML (from [LLZ+14]). Yellow is for the core module of IndoorGML, and orange for OGC Geography Markup Language Encoding Standard (GML) [PCD+07].

Compared to CityGML and IFC, the IndoorGML is mainly about the description and representation of indoor navigation networks and spaces. IndoorGML aims to define the indoor space instead of building features such as in CityGML [RKL15]. The space classes of IndoorGML are relevant to elements of navigation networks. These networks are designed according to Poincaré Duality [Whi32, Mun84a]. Duality is a one-to-one mapping relationship between two related geometries. A planar graph [Whi32] consisting of nodes and edges can be formed on the basis of a 3D room (i.e., a 3D closed manifold): one node for each room and one edge for every two rooms with adjacent
faces. In this case, a node is the dual of a room and this graph is called a dual graph. The vector space regarding all these original geometry (rooms) is named primal space, while the vector space containing all these duals is named dual space. Although the IndoorGML provides a schema framework for indoor navigation based on space connectivity, it does not introduce the transformation method for the network from building data.

Different navigation networks can be integrated into the multilayered space-event model (MLSEM) [BNK09]. As the ‘event’ represents dynamic information such as leaving or entering a room, here only the ‘space’ part of MLSEM is discussed. The MLSEM provides a multilayer representation for different spatial models, such as the topographic space for 3D buildings and the sensor space for sensor range partition. Buildings can be subdivided not only with respect to the topographic /geometric / construction properties of buildings, but also regarding the spaces defined by security reasons, Wi-Fi coverage, motion-impaired users, emergency cases, etc. (Figure 2.6b). In the MLSEM model, each space layer is mapped into the primal and the dual spaces according to Poincaré Duality. For example, a 3D room in the primal space corresponds to a node (0D) in the dual space. On another dimension, each space layer is also divided into the topology and geometry spaces (Figure 2.6a). Various layers of the space models are connected by so-called joint state edges, which represent the space overlap of two nodes from the two space models. At any one time only one joint state edge and related nodes are active. In the dual space of topographic space layer (Figure 2.6c), navigation networks can be derived in the same way as the one reported by Lee [Lee04].

A building ontology refers to the semantics of indoor spaces, and the semantics can be applied to facilitate the generation of navigation models of a building for pedestrians (usually graph structures). An ontology describes a set of definitions of classes and properties and their relationships for a particular domain [NM01, BCC06]. For instance, it is relatively straightforward to obtain connectivity relations between indoor subspaces with knowledge of doors and rooms. Furthermore, navigation-related semantics of indoors (e.g., navigable space, obstacle, etc.) facilitates routing, and user-related semantics (mobility, transport preferences, etc.) are more perceivable to users. Based on American Disability Act standards, Dudas et al. [DGK09a] develop an ontology and an algorithm named ONALIN, which considers the needs of different groups and individuals on their feasible routes. Karimi and Ghaforian [KG10] propose ontologies about path segments and points of interest which aim to provide safe passage for the visually impaired. Tsetsos et al. [TAKH06] give an ontology of building elements and paths, and a comprehensive list of user modelling as a user ontology. Goetz and Zipf (2011) present another ontology named 3D Building Ontology (3DBO) (see Figure 2.7) about building elements and navigable parts [GZ11a]. But these semantics are either too general from a navigational point of view [TAKH06], or too specific separating similar spaces such as a room and a corridor [GZ11a].
FIGURE 2.6 MLSEM layers represented in different space forms. (a) Different forms of building representations, (b) The building representation in primal topographic space, (c) The derived dual graphs (i.e., networks) (from [BNK09]).
FIGURE 2.7 The 3D Building Ontology (from [GZ11a]). It mainly describes the inside of buildings, and also presents navigable parts such as classes VerticalPassage and HorizontalPassage.
To obtain the semantics of spaces, a building needs to comply with a certain subdivision \cite{ZLS13}. Different subdivision results consist of distinct types of space on functionalities. For instance, a lobby can be separated into more spaces, or it can be seen as one. In some cases, source building models have provided a subdivision result. Various semantic data models have been developed with the focus of users or easy extraction of navigation networks \cite{DGK09b, KG10, Wor11, YW11}.

A subdivision concentrates on the minimal space unit with close space size to be identified. Generally, two types of subdivision can be distinguished – structural and functional subdivisions \cite{RWS11}. A structural subdivision follows the physical structure of a building (e.g., an office bounded by walls), while the functional subdivision defines spaces according to their functionalities, and provides comfort, safety and security to ensure the necessary boundaries of separated indoor spaces \cite{KZ14}. For instance, Richter et al. \cite{RWS11} separate rooms into offices, laboratories, computer rooms, facilities (e.g., toilets), passages (e.g., corridors), etc. The functional subdivision also aims for different users \cite{TAK05, RWS11}. Kruminaite and Zlatanova \cite{KZ14} extend the functional subdivision method to consider functional subspaces (which may be inside of a larger space) of indoor objects, depending on their characteristics such as attractiveness, necessity, limited capacity, closeness to central areas, and possession of transition area.

The subdivided spaces refer to the nodes of the topological model (graphs) of a building. But for one set of nodes, various relationships can be established. For example, a connectivity graph provides space relationships that indicates an agent can pass from one subspace to another. An adjacency graph denotes all the neighbours of a specific space. Furthermore, not all spaces might be considered (or accessible) in a specific navigation case, which results in another type of topological model, i.e., the accessibility graph \cite{Wor11}. Thus, a topological model heavily depends on the subdivision result and the relationships between these spaces (i.e., edges in the graph) \cite{NSK09}. These topological models (i.e., a kind of navigation models) will be further discussed in the next section.

Navigation models can be derived from building models for pedestrians. A navigation model is the computational foundation for routing, such as navigation networks or grid models. Indoor routing can be conducted on the navigation models. Previous work shows that 2D geometry, such as floor plans, is frequently used to generate adjacency and connectivity networks \cite{PZ05, LOS06b, SLO07, MZLC14}. There are approaches regarding 3D building models as well but they are either based on 2D floor plans \cite{JTY11}, specific application \cite{SZVO11}, or at quite a conceptual level \cite{BNK09, BNZK13}. In these approaches 3D building models are mainly used for visualization, after the path is computed based on 2D floor plans. The next section will introduce different types of navigation model.

\section{Navigation models}

Indoor navigation models represent the interior of buildings in a simplified way, but they contain sufficient information for conducting navigation tasks. Two types of model can be distinguished, that is, network (vector) and grid (raster). Network models are
more widely used in pedestrian indoor navigation, while grid models are predominant in robot navigation. Commonly topological relationships, geometric information and semantics have been employed for indoor navigation on network models [Wor11]. The details of topology, geometry or semantics represented in the reported network models differ significantly. There are two big groups of network models, i.e., models that preserve the geometry of the building [JM05, LOS06a, MZP05, PZ05, SLO07] and those omitting geometry [BD05, BS01, FMB00, GSC05, HD04, LL08, JS02, RWS11, SSO08, YCDN07, HOP08]. The first group of models are more suitable for visualizing paths since the paths include accurate geometric shapes inside the building. The second group of models results in more compact representations that are very convenient for conceptual analysis.

In 2D building models, navigation models are mainly derived from two types of subdivision: subdivision according to a certain criterion and regular subdivision. Some of these models adopt semantics (such as notations of doors, windows, walls) to refine navigation paths. The subdivision according to a criterion can generate navigable spaces from floor plans by following the building structure, or can break down 2D floor plans into cells according to certain criteria (e.g., convexity, visibility, max cell size, functionality, etc.). The regular subdivision results in regular grids such as rectangular, hexagon, octagon, etc., which represent the spaces at a certain granularity [ICC12].

In 3D building models, related research [Lee04, MZP05, JTY11, HEZ12, CWSC14] mostly places a 3D representation by the layered model of 2D floor plans. The regular subdivision results in navigation networks based on the building structure, while the subdivision according to a certain criterion can generate either networks or regular grids according to a specific partitioning (e.g., constrained Delaunay triangulation algorithm or visibility criterion). To simplify the complexity of 3D geometry, semantics are also being largely incorporated into these navigation models. In the following parts navigation networks and grids are introduced for both 2D and 3D building models. In both 2D and 3D building models, subdivision can result in different hierarchies of indoor space, i.e., multiple levels of indoor space such as floor, section, room, and subroom. Some examples of such space hierarchy will be presented later in this section.

FIGURE 2.8 A 2D network created on the basis of Poincare Duality, Media Axis Transform and information about doors. (a) Floor plan; (b) The metric network based on connectivity of spaces; and (c) The metric network considering door locations (from [MZP05]). The nodes cr32, cr0918, cr0819, cr0720, cr0106, cr42 represent precise geometric locations in the space related to node r60 (in subfigure (b)).
2D Navigation Network. Usually the network is based on Dual Graph, Media Axis Transformation (MAT)/centerline/shape skeleton algorithms, Visibility Graph (VG) or combinations of them [ICC12]. Figure 2.8 illustrates the most common approach utilizing a dual graph, MAT and information about doors (i.e., straight MAT) [MZP05, CL09]. The dual graph [Whi32, Mun84b] results in the room-to-room type of paths since each room is represented by a node. If the MAT method cannot result in a sufficiently detailed path, new nodes are introduced to provide semantics, i.e., building elements such as doors and windows (Figure 2.8c). Mortari et al. [MZLC14] propose another network model based on Constrained Delaunay Triangulation to improve the MAT-style methods. This network is generated with consideration of space between indoor obstacles. Such a network would be re-computed if indoor obstacles change. Besides, many studies have discussed topological and semantic representations of indoor spaces for both robots and pedestrians [BS01, GSC+05, JS02, RWS11].

Another approach to create a navigation network is the VG method [Lat91, dBCvKO08]. Some research of indoor pedestrian navigation straightforwardly employs VG algorithms or slightly modifies them for certain purposes [HBK+10, KBH12, SGS12]. In contrast to MAT, VG does not follow the shape of the building spaces (Figure 2.9), but provides a direct path among points of interest, i.e., the door-to-door type of paths [LZ11a]. Commonly space subdivision is needed in this case since VG is constructed inside each room. VG networks also need re-computation if there are changes of indoor obstacles (e.g., with extended size).

Lorenz et al. (2006) proposes a modification of the MAT- and VG-based approaches. In this case doors are abstracted as nodes (i.e., they are considered spaces, see Figure 2.10), and a room can be represented by only one node (representing a small space) or several ones (representing a large space, such as nodes cr32, cr0918, cr0819, cr0720, cr0106, cr42 in Figure 2.8c). This approach allows for the room-door-room type of paths.

Some other methods consider the subdivision of a 2D floor into cells according to certain criteria. Several typical criteria to subdivide indoor spaces are found in the literature. To help robots pass through bottlenecks and avoid collision between moving objects, Lamarche and Donikian [LD04] apply a series of algorithms such as the constrained Delaunay triangulation algorithm and Convex Cell optimization (i.e., to merge the resulting triangles into convex cells which contain the minimum number of cells) to subdivide 2D floors (Figure 2.11). The original floor is subdivided into smaller convex
regions, which can be used to derive a navigation network. Stoffel et al. [SLO07] propose a method that partitions a floor plan into convex regions according to the visibility criterion (see Figure 2.12). In such a convex region, openings are mutually visible. This is a typical example of the door-to-door (or portal-to-portal) navigation.

According to Xu et al. [XWZ16], the MAT method cannot ideally deal with large complex spaces, and VG-based methods need to use obstacle vertices as nodes for navigation networks. They propose a subdivision method for 2D floor plans based on Delaunay Triangulation which can generate a network inside a room and passes through gaps of obstacles. However, the time complexity of this method would need to be further clarified if it was to be applied to large complex buildings.

Wallgrün [Wal04] develops a method for robot navigation based on the generalized Voronoi diagram. This approach generates navigation networks relying on the pure geometry (including obstacles) (see Figure 2.13). In this case notations of doors are of no importance. In addition, some nodes at corners of the network are removed (Figure 2.13c).

3D Navigation Network. Researchers [Lee01, Lee04, MJ05, PZ05, BG10] generally classify the 3D geometric models of buildings into the geometric network and the topological network that represents the connections among spaces in buildings. The topological network is used to compute conceptual paths and the geometric network is used for accurate routing and visualization. In order to conduct reliable and fast computation, many researchers adopt the graph model to represent connectivity relationships of indoor spaces [Lee04, MJ05, LD04]. In the geometric network, detailed paths can be computed more accurately for pedestrians.
FIGURE 2.11 The subdivision with Delaunay triangulation and Convex Cell optimization (from [LD04]).

FIGURE 2.12 Visibility partitioning result (from [SLO07]).

FIGURE 2.13 The navigation network derived from the Voronoi diagram. (a) The original plan; (b) The navigation network regarding spaces on a floor; and (c) The simplified network in the higher hierarchical level, i.e., floors (from [Wal04]).
FIGURE 2.14  An example of the navigation network in 3D. (a) Floor plans representing indoor spaces; (b) The topological network (connectivity graph) of these indoor spaces; and (c) The geometric network. The long spaces S6 and S12 are transformed to more refined nodes, and edges are precise paths between two nodes (from [Lee04]).
Lee [Lee04] designs a *Node-Relation structure* (NRS) (i.e., *Dual Graph*, see Figure 2.14b) to represent the connectivity of buildings [Lee04]. *Room-Door* relations are converted in the primal space to *Node-Edge* relations in the dual space [Whi32, Mun84a]. In order to represent indoor environments more accurately, Lee (2004) extends the NRS to the *Geometric Network Model* (GNM), which introduces metrics [Lee04]. Lee (2004) also adopts a skeleton-abstraction algorithm to construct a 3D GNM (Figure 2.14c), i.e., *Straight-Medial Axis Transformation (S-MAT)* modelling method [EE99, CL09], which can abstract linear features from simple polygons (such as corridors).

![Figure 2.15](image)

**FIGURE 2.15** 2D floor plans embedded in 3D space and linked to the outdoor network (from [JTY11]).

In some cases (especially within regular buildings), 2D floor plans are embedded into 3D spaces of a building. Thill et al. [JTY11] adopt this approach and demonstrate it can be applied in the combination with outdoor networks (Figure 2.15). Such approaches manage to accommodate many properties specific to indoor spaces, such as ingresses and egresses, vertical movements in stairs and elevators, movements on escalators, and segments that are not accessible due to impaired motion ability. The resulting visualization and rendering can be achieved in both 2D and 3D views and can enhance the guidance associated with individual movements through indoor spaces. Navigation on the individual 2D floor plans can be performed according to any of the 2D approaches mentioned in the previous section. However, such 3D cases need information about walls and ceilings or more detailed locations in vertical connecting spaces such as stairs and elevators.

Another group of approaches consider walkable connected surfaces for navigation [Sli06, SR08, Sch10, SZVO11], without explicit networks. In this case, topologically-connected and navigable spaces (surfaces) are embedded in 3D space. Slingsby and Raper [SR08] construct a navigable space model from 2D plans with additional information on heights and surface constraints (Figure 2.16). This approach ensures the connectivity relationships among spaces (represented by these surfaces).

Another type of network can be built with volumes and surfaces: the dual of a volume is a node and the dual of a surface is an edge (Figure 2.17a). Boguslawski et al. [BGL11] follow a data structure similar to the NRS structure [Lee04] (Figure 2.17c). There is little elaboration on the semantics but it is assumed that the approach would work with any space subdivision and space definition, which can easily generate topological networks.
3D semantic models of buildings also support the generation of navigation networks, such as CityGML LOD4 and IFC. The inherent subdivisions of the semantic models can be used to derive the connectivity via openings (e.g., doors and windows). Theoretically CityGML LOD4 [GKNH12] offers a straightforward approach as the rooms are described by bounding surfaces that link to openings (Figure 2.18). Information about obstacles can be derived from the objects defined as IntBuildingInstallation. However, other obstacles such as moving objects cannot be mapped from CityGML. Liu and Zlatanova [LZ13a] have presented the generation of topological networks from a CityGML LOD4 model of a building (see Figure 2.19).

There is similar research on creating networks based on IFC models [LH08, TC16]. To derive a network from an IFC model, Teo and Cho [TC16] propose a network called a multi-purpose geometric network model (MGNM) (including indoor routing, see Figure 2.20). The MGNM is built in the following steps: 1) extract building information from IFC, (2) create the MGNM (i.e., a geometric network) on the extracted information, and (3) create topological relationships of the MGNM in a Geodatabase.

In a full 3D model, Diakité and Zlatanova [DZ16] propose a method to extract the navigable space considering indoor furniture. These semantic spaces are created for indoor routing, such as a network of connected 3D spaces (volumes). With such spaces, one can know the exact extent where a subject (a pedestrian, robot or drone) can navigate.
Boguslawski et al. [BMZF16] propose a construction method about *Variable Density Network (VDN)* to determine egress paths in dangerous environments, which includes a full 3D topological model. They consider VDN can increase the accuracy of prediction for egress path planning.

There is a special combination of navigation networks – hierarchical graphs. Hierarchical graphs [BD05, BS01, HD04, JS02, JM05, LOS06a, SSO08, YCDN07] contain topological and/or geometric networks of spaces (*e.g.*, *floors*, *zones*, *rooms*, and *subspaces*). Figure 2.21 illustrates such a model. The two floors include fifteen rooms and two subspaces in the room r14 (see Figure 2.21a). On the floor level, the topological network of the floors is represented by two nodes f1 and f2 (see Figure 2.21b); on the room level, the two floor nodes are extended into two topological networks of rooms. On the sub-room level, the room node of r14 is extended into a topological network of two subspace nodes (sr1 and sr2). In these models, distinct topological networks can be derived based on different space decompositions of a building.

The hierarchical graphs aim to represent a complex environment with a spatial representation of multiple graphs. Normally navigation networks are of a small size com-
pared to road networks, which indicates the topological network of buildings is generally small. Thus, a user does not need a hierarchical graph in a simple building when routing can be easily achieved.

**2D Grids.** Another large group of approaches is based on regular grids such as rectangles, hexagons, octagons, etc. [ICC12]. The discrete 2D grids overlap the 2D plan (Figure 2.22). Each grid cell obtains semantics according to the underlying 2D objects (rooms or doors). This approach allows for a precise localization in a large open space, and it is often used for applications which require tracking and correction of positions [GCZ+ 11] or integration with continual phenomena such as smoke or fire. The grid-based approaches originate from robot navigation. Grids allow incremental movement (and speed control), which facilitates driving, collision detection and manoeuvring of robots. The size of the grid is of critical importance. If the grid size is too coarse, important indoor information may be lost. Alternatively, too fine a grid may increase the computational load.
FIGURE 2.20 The network created from a BIM model (from [TC16]).

FIGURE 2.21 The hierarchy graphs for two floors. (a) Two floors of fifteen rooms with two subspaces; and (b) The hierarchy of floors, rooms, and subspaces. The hierarchy includes 2 floor nodes, 15 room nodes, and 2 subspace nodes.

FIGURE 2.22 The subdivision of square and hexagon grids (from [ICC12]).
FIGURE 2.23  An example of a 2D grid model. (a) Semantic annotations of a floor, (b) Generated grids, and (c) The shortest path on the network derived from the grids (from [LCR10]).

FIGURE 2.24  24 and 18 search directions of nodes on the grid model (from [VBWVHVO93]).
A network can be generated from the grids, and the navigation network of the grid model can be implicit and avoid storage of many grid cells with \( O(n^2) \) space complexity. For example, the nodes represent the centres of the grid cells and the edges of a certain node represent connections between the node and its neighbours [LCR10] (Figure 2.23). Moreover, the movement of a grid can be planned in multiple numbers of direction. Bemmelen et al. [VBWWVHO93] propose an approach to make more search directions for moving through a raster cell (grid, see Figure 2.24). The overview of the 2D grid approaches does not aim to be complete but contains only the principle tendencies. As grid-based approaches mainly rely on geometric subdivision (i.e., a continual space into grids), the related path finding on the grid models does not heavily rely on the topological and functional meaning of indoor spaces.

3D grids (voxels). Basically, 3D grid approaches are an extension of 2D grid ones. A 3D grid-based (voxel) network can represent the 3D structure of indoor space. For example, intermediate levels in a floor (e.g., ramps and platforms in the air) can be represented. Yuan and Schneider [YS11] propose a model called a LEGO graph based on 3D voxels (see Figure 2.25). This method computes the accessible parts of indoor environments and considers the constraints of the width and height of users (such as drones).

![Figure 2.25](image) The LEGO graph. (a) Subdivided blocks of a floor; (b) The graph representing the connectivity of blocks; and (c) The resulting LEGO graph (from [YS11]).

Bandi and Thalmann [BT98] discretize the whole scenario space into 3D voxels and compute an obstacle-free feasible route with consideration of surmountable and insurmountable obstacles and the 'hole area' (i.e., insurmountable obstacles may be encompassed by a closure of reachable grids) in buildings (Figure 2.26). This method can easily deal with height information and can generate routes of various heights (Figure 2.26b).

3D voxel models (see Figure 2.27) for navigation are also applied to game simulations [Hel13, GBK14]. Game simulations have many similarities with robot and human navigation. Many agents (game characters) navigate similar to humans. For example, they can jump over the low obstacles or even get through the holes in walls if allowed. A typical study in this field is to project all the obstacles of the animated scene to a 2D bitmap to accelerate the process of forming a path plan [KJ98]. Andújar et al. [AVF04] present an algorithm for camera path computation based on voxelisation. This algorithm determines the free-space structure of the scene and provides various measures to find out the best path for visiting (Figure 2.28).
FIGURE 2.26 Reachable voxels and the generated path (from [BT98]).

FIGURE 2.27 An example of a voxel model (from [Zha13]).
FIGURE 2.28  Top view of the original model; high-level path through the five most interesting cells (the dashed lines) and the computed low-level path (from [AVF04]).
Generally, 3D regular grids (voxels) have a great potential for indoor navigation. The voxel model can readily indicate the membership (e.g., corridors, rooms and stairways) of each voxel and incorporate various semantics of these voxels. 3D voxels can also take into consideration height constraints and allow path computation at certain heights (i.e., flying), under or around objects by applying the shortest path algorithms such as A-star [HNR68] (see Figure 2.27).

Fichtner [Fic16] proposes a workflow for semantic classification based on unstructured 3D indoor points by using an Octree structure. With some preconditions (e.g., wall perpendicular to floor), indoor semantics such as floor, storey, stair, and wall can be identified. The walkable surface consists of floors and stairs. This work presents the potential of the semantic-enriched data for indoor pathfinding based on walkable voxels. But the process of identifying stairs needs to be improved for other applications.

Regular grid methods with an appropriate resolution are slightly superior to network methods on locating agents, because they are mostly developed for the more complex robot navigation. Network methods tend to be used for human navigation and as such, they provide less detailed routing, assuming that human intelligence will compensate for inaccuracies. Human-based network routing considers semantics much more in comparison to grid-based routing. The network always contains the functional or thematic meaning of indoor spaces, while a grid model considers space semantics less.

§ 2.3 Routing algorithms and methods

In this thesis, routing is referred to as the automation of searching path on a data structure (e.g., graph). Single-source path finding is the focus in this thesis, i.e., one user or a small group of users moves from a start location to one other place (or multiple places). In fact, this routing is the single-source shortest path problem. A number of algorithms has been developed to solve the problem such as Dijkstra [Dij59], Bellman–Ford [FJ56, Bel58] and A-Star (A*) [HNR68]. The Dijkstra algorithm finds the shortest paths from a source node to all the other nodes in a graph. The Bellman–Ford algorithm also solves the single-source problem when edge weights of a graph are negative. The A* algorithm adopts heuristics to speed up the search for the shortest path between a pair of nodes. They are algorithms run on a graph which is referred to as a navigation network in our context. For 2D/3D grids, these algorithms can be conducted on the graph that represents the link relationships of these grids. Some heuristic algorithms such as the D* (a dynamic A*) algorithm [Ste94, Ste95] are also developed for robot navigation in dynamic environments.

Other graph-search algorithms such as breadth-first search (BFS) [Lee61] and depth-first search (DFS) [Sed02] can also aid pathfinding in buildings, such as in hierarchical graphs. Specifically, the BFS is a special variant of A*. Bellman [Bel58] introduces a dynamic programming approach to solve the minimum travel time problem. In this problem, an optimal path (whose cost is travel time) through multiple locations linked by road should be determined. Besides, only a finite number of iterations will be required in this iterative algorithm. This method can be applied to navigation networks to compute indoor paths.
Although the above algorithm can be applied to navigation models (networks or grids) of buildings and compute paths for indoor navigation, many studies propose ad-hoc routing methods and routing criteria relying on specific navigation models and user needs [WMY07, LSA08, CDO08]. Different from outdoor navigation, indoor paths are relatively limited and the distinct requirements of various users are significant for indoor navigation. For example, the shortest path can be readily computed in a navigation network of a building, but it may not be the ‘optimal’ path according to the user (e.g., ‘optimal’ refers to the fastest path). As the ‘optimal’ definition can vary, the focus of indoor routing is to obtain different indoor paths according to given criteria.

Indoor routing for pedestrians needs a larger spectrum of paths compared to outdoor. The criteria for outdoor routing mostly are distance, travel time or number of turns [DGK09a] that are all based on metric information (e.g., length and angles). Visser [Vis09] proposes a path-finding approach which concerns changes in road environments and predicts future situations. This approach can automatically generate routes within disaster areas with consideration of the changing gas plume and temporarily closed roads. But this method is only applied in a 2D road network. Another alternative is the Indicative Route Method (IRM) proposed by Karamouzas et al. [KGO10]. It is built on the corridor map proposed by Geraerts and Overmars [GO07]. The corridor map structure offers a set of collision-free spaces. The IRM can generate routes as smooth skeletons in a corridor map. Though IRM is a convenient method for considering obstacle avoidance, currently the network of indicative route cannot be automatically created.

For indoor navigation, there are many non-metric factors influencing indoor routing (e.g., cognitive similarity to pedestrians, temperature deviation of spaces, the number of visual signs). The mostly frequently used criteria for routing are geometric [GZ13, MZP05, Whi06, YS11]. Dudas et al. [DGK09a] define two other types of paths: feasible and comfortable paths. The term feasible refers to accessible paths for users with special needs (e.g., mobility-impaired), while the term comfortable indicates the subset of feasible paths that is assumed to be preferable to the users. Each edge of a geometric network is assigned a weight representing the degree of comfort to a user. The degree of comfort is specified according to the user’s preferences. The comfortable path is a user-specific path especially about mobility (e.g., elevators for motion-impaired users). For a set destination regardless of their order, the comfortable path is the one through these destinations with the minimum cost. Then the comfortable path can be computed by shortest-path algorithms based on the weights. In city scenarios, the least visible path [LZLF08] is proposed to search a least-cost path which stands for the least chance to be sighted. This routing criterion uses reverse viewshed at each location (i.e., the visible area viewed from the other locations). The resulting least-visible path includes the minimal visibility privilege.

Another non-geometric criterion is the Least-effort [CWSC14] which minimizes the total required travel time to reach a destination. In indoor environments, time and safety are vital factors in emergencies [PZ05]. Moreover, environment, event and human factors should be considered for indoor path planning [ZB08]. For instance, crowd flow velocity changes over time on connectors (corridors and stairways) of buildings. In addition, individual parameters (e.g., physical ability) critically limit indoor path selection as well.

Li and Lee propose [DL08] the notion of semantic distance including geometric distance information and the graph structure information between two locations. It con-
nects to the number of spaces (e.g., the number of connective nodes in a path) and the number of doors in between two spaces. In a connectivity graph of indoor spaces with edge weights of semantic distance, graph-based algorithms can be applied to path computation. Another useful criterion for a topological network is centrality. For example, in a connectivity graph or adjacency graph of a building the centrality of a node measures the relative importance of the node regarding accessibility [S.05]. The centrality can be computed in different ways such as the degree, betweenness and closeness. The degree of nodes indicates the number of nodes connecting to a node. The closeness measures how ‘close’ a node is to the others. The betweenness of a node is the ratio of the number of shortest paths via the node and that of all the shortest paths (among all the possible pairs of the start and target nodes).

Routing with multiple criteria is seldom discussed in indoor routing. This type of routing is a kind of multi-objective optimization [CDO08, Deb14, JT10], which aims to solve the optimization problem regarding a number of (even contradictory) objectives. Some related work is found in evacuation routing research, which simulates the behaviour of dense crowds. Multi-objective optimization approaches have been considered as suitable ways to address the realistic requirements of evacuation planning [AS09, SS09, LLHY08]. In the hierarchical directed network of a stadium, Fang et al. [FZL+11] propose a multi-objective optimization approach to solve the evacuation routing problem. Three criteria are considered in this approach, i.e., minimal evacuation time, minimum of total evacuation distance and minimal cumulative congestion degrees. This approach is based on the ant colony optimization (ACO) algorithm [BM04, Blu05] to simulate the evacuation process.

Lyardet et al. [LSA08] compute indoor paths for users according to multiple factors in addition to distance, such as temperature, crowdedness, and turns. This path computation employs a weighted-sum method [Deb14] to compute a unique path from a set of feasible paths. As a typical example in outdoor networks, Nadi and Delavar [ND11] propose a general approach for multi-criteria and personalized path planning (regarding user preferences), which results in multiple alternative routes from different decision strategies.

Another easily conducted method, Lexicographical Goal Programming, is developed in the research of multi-objective optimization [CDO08, Deb14, JT10]. When there are several different criteria to be selected, preferences have to be set up (the importance of the criteria). The method has been applied to routing with priority of criteria. A user has to specify the order of the routing criteria. Paths are computed according to the first criterion (i.e., the most important one). Then the computed paths form a new smaller network and new paths can be derived on the new network in light of the second important criterion, and so on. The computation stops when all the criteria have been checked or only one path is found.

Apart from the above routing approaches regarding different criteria, there are other indoor routing methods relying on specific navigation models or structures. Wu et al. [WMY07] propose a path-planning algorithm within indoor environments for visually impaired. This method consists of three parts: cell decomposition, cactus tree-based path planning for the building, floor and area, and the A* based path. The cactus tree (See Figure 2.29) is a non-linear data structure of relationships among indoor elements (the building, floor regions, and locations) for path searching [WMY07]. Rela-
tionships in this tree are hierarchical, and a cactus tree-based search can be used in path planning in a floor, area or between regions.

Moreover, another type of solution has been proposed as the triangulation-based method [BS01, CDO08, VTCG11]. Based on a particular triangulation strategy, indoor environments are represented by different triangular areas. Thus a certain MAT network can be derived and used for indoor routing.

Rodenberg et al. [RVZ16] present an indoor routing case based on an Octree structure derived from point clouds of a building. This research generates an Octree of 3D voxels for a building from point clouds, and then forms a navigation graph by considering the connectivity for each node to all its possible neighbours (e.g., other faces, edges and vertices). Finally, the A* algorithm is applied to the derived network for pathfinding.
FIGURE 2.30 The path computed by the corridor map algorithm. (a) Corridor map with the points of the closest distance; (b) The closest points along a route and corresponding clearance; (c) Shrunk corridor; (d) Triangulation; (e) Funnel shortest path algorithm; and (f) The smoothed path (from [R.10]).
Geraerts [R.10] presents an approach for path computation by delineating the space, navigating through the spaces in a given tolerance and then smoothing the path (Figure 2.30). The result is equivalent to the VG-based path computation, but avoids the complexity of creating VGs for the entire indoor environment. This approach exclusively relies on geometric algorithms, especially for avoiding obstacles. Though this method is applied to outdoor routing, it can be transplanted into the indoors when 2D floor plans are available and the status of all doors is considered as ‘open’.

Xiong et al. [XZZ+15] propose a new type of indoor routing method to make use of semantics. This method voxelizes indoor scenarios, derives regions with navigable area and semantics, traces boundary of regions, and then generates navigation meshes. Routing on these meshes provides a path between two semantic locations. In the resulting paths, some of them are close to walls and stairs, which needs to be improved.

There are also routing computations regarding hierarchical graphs/networks of buildings. Hu and Lee [HD04] propose a hierarchical symbolic model, exit hierarchy and location hierarchy based on a tree model, where the shortest path algorithm can be applied for routing. A number of studies [JM05, TAK+05, YCDN07, GSC+05, HOP+08, SSO08, RWS11] propose hierarchical networks of buildings where routing can be performed by using common path-finding algorithms (e.g., the shortest path algorithms).

**FIGURE 2.31** Minkowski sums of obstacles to a user and the minimum distance between obstacles. (a) Minkowski sum of obstacles for a user approximated as a circle; and (b) Union of the Minkowski sum of obstacles and the minimum distance.

Another important issue for indoor pedestrian navigation is obstacle avoidance. Previous research has considered indoor obstacles in the navigation network [GZ11a, LYJS09, SR09, MZLC14] and obstacle-avoiding path finding [HBK+10, KBH12, SGS12]. However, the work has not discussed the influence of user sizes. In contrast, robot motion planning has always taken into consideration the dimensions of robots. The Minkowski sum method [dBCvKO08, Coe12] has been commonly applied to identify inaccessible areas for a robot. Figure 2.31 presents an example where a robot is approximated with a circle. A Minkowski sum of an obstacle expands the obstacle according to the robot’s size (the circle’s radius in Figure 2.31a), while simultaneously the robot shrinks to a ‘reference’ point (see Figure 2.31a). The Minkowski sum of obstacles represents the inaccessible area for the robot. If the Minkowski sums of different obstacles intersect, then they will be merged into one to form a closed area that is inaccessible for the robot (see Figure 2.31b). The other space is regarded as free space for the robot, and the robot can follow paths in it.
There are a few studies discussing the dimensions of pedestrians for indoor routing. Generally, navigation models for pedestrians do not refer to accessible indoor areas of users [CWSC14, HBK+10, KBH12, Lee04, LOS06b, MZP05, SGS12, SR09, JTY11]. They implicitly regard a user as a point or approximate the user with a very small size. Yuan and Schneider [YS11] model indoor space with different types of cubes and merge the cubes to reflect the accessibility for users. However, the study did not provide a detailed or practical solution to computing paths for users with different dimensions.

Mostly, the size of users has been taken into account to investigate the accessibility of indoor environments for wheelchair users [HLLK02, KS15, OMP09, Pru10]. Han et al. [HLLK02] employ the Minkowski sum method to outline the accessible areas for wheelchair users. Otmani et al. [OMP09] and Pruski [Pru10] pinpoint the accessible areas for wheelchair users with respect to the orientation of the user. Their approach is also based on the Minkowski sum. Kostic and Scheider [KS15] propose an approach for computing accessible areas on a grid model (i.e., regular cells). According to the shape of the user and the wheelchair, the computed areas can support the movements aligning with the x- and y-axis and the 90-degree rotation case. All the research aims to find the bounded polygonal/grid accessible areas for wheelchair users and then to compute paths inside the areas.

§ 2.4 Pedestrian wayfinding behaviours

Indoor navigation involves both guidance (i.e., instructions/indications for how to physically move in computed paths) and pathfinding (i.e., to compute paths between or among different locations). The related notion ‘wayfinding’ is about how pedestrian themselves find their way in a cognitive process [HB15]. Navigation guidance can be reflected by path instruction, i.e., detailed explanation of a computed path for pedestrian users. As addressed in Chapter 1, this research does not focus on indoor guidance. Wayfinding is about the process that a pedestrian employs to independently find her/his way with aids of indoor configuration (e.g., signs). Thus, wayfinding is helpful to explore pedestrian behaviour modes and investigate path selection criteria.

Wayfinding for outdoor environments has been studied for decades. The well-known landmark-route-survey framework [SW75, TG83] indicates the spatial knowledge representation in human cognitive processes. The knowledge provides people with self-guidance to a target location in an environment: he/she can use landmarks to orient herself/himself to the destination, follow a continuous route, or complete their orientation by survey knowledge (the complete cognitive image of the environment). Thorndyke and Hayes-Roth [THR82] point out that maps provide a user with general survey knowledge rather than precise route knowledge. Maps may also interrupt the orientation of a user since the user cannot align her/his motion with the maps [LJP82, BAHS93]. Therefore, many studies focus on making use of landmarks [RW02, MS07, XAJC08, HGL+09].

In a similar way, wayfinding also supports a user navigating inside buildings [MZV15]. Soeda et al. [SKO97] include vertical motions in the wayfinding process, which importantly influences the movement among different floors. Hölscher and Brösamle [HB07] show that a user unfamiliar with a building has more difficulty in performing wayfind-
A complex building causes a heavy memory load even for a user acquainted with the building. In this case, direction is significant during the wayfinding [KT03, PH12]. A user sensitive to directions presents better performance than those who are not sensitive. Meanwhile, the user with a better sense of direction tends to flexibly use effective strategies related to turns, directions, and landmarks. However, indoor navigation (including wayfinding) is commonly developed to navigate users unfamiliar with a complex building. In addition, a sufficient navigation system should support common users, no matter whether they have a poor or keen sense of direction. For example, Stook [Sto12] develops a solution to transform indoor Wi-Fi-based positioning results to location information (e.g., Room 2.200). In this way, Stook [Sto12] forms clear descriptions about paths for users according to their profiles.

The solution of using landmarks can compensate for the disorientation of a user. People can adopt landmarks as anchors and conduct wayfinding without better survey knowledge [SK07]. Similarly, indoor wayfinding values landmarks since they reduce the memory burden of a user. Additionally, using landmarks benefits the generation of path instruction [HP10]. Thus, landmarks are ideal tools for indoor wayfinding since they support effective interactions between humans and environments [RW14]. Landmarks have also been adopted to generate path instruction for guidance, such as Walk to the lower end of the stairs marked with the sign ‘Neubaugasse’; then walk up the stairs [RGL+07]. In general, landmarks make the wayfinding more tangible to pedestrians.

The indoor wayfinding research inspires some strategies for indoor navigation. There are some heuristic methods aiming to simplify as much as possible a wayfinding process [CTG+06, HB07]. Hölscher et al. [CTG+06] differentiate three wayfinding strategies, i.e., the central point strategy, direction strategy, and floor strategy. Using the central point strategy is to find a path by trying to transit well-known locations (e.g., landmarks) of buildings. By applying the direction strategy, a user heads to the horizontal position of a destination as directly as possible and regardless of the level changes. By applying the floor strategy, a user needs firstly to find a path to the storey of a destination, then horizontally move to the destination on the storey. These strategies reflect wayfinding behaviours. Some other factors such as the colour and light of an indoor environment can also influence wayfinding performance [HYA12]. In addition, Rüetschi [Rüe07] indicates that structural information of buildings can support wayfinding, though it may not result in optimal paths. In general, all the research findings can be adopted to design and compute indoor paths imitating the wayfinding behaviours.

§ 2.5 Indoor positioning and tracking

Nowadays indoor navigation heavily relies on an accurate and stable positioning or localization technique. Unfortunately, most existing positioning techniques are still at an experimental phase [Mii06, CNPM11]. Compared to outdoor GPS tracks (recordings of positions at regular intervals), indoor positioning suffers from low accuracy, which results in a limited number of indoor tracking applications. Current types of localization systems are based on different techniques, including Angulation (angle), Lateration (distance), Fingerprinting, Inertial and motion sensors, and Neighborhood [CNPM11]. Here the Fingerprinting method is the focus since it is the currently popular method.
for indoor pedestrian localization and some commercial applications have been developed.

To be able to localize a person or robot in a given indoor area, the indoor space has to be partitioned. Such artificial subdivision/decomposition can be based on a regular grid (grid for short), triangulation tessellation, trapezoidal-based tessellation and Voronoi diagrams [ICC12]. Grid is widely applied to indoor navigation and tracking. Li et al. [LCR10] elaborate on a grid graph model. They first overlay the building parts/ cellular units (such as a room, a wall, etc.) with grids and then generate a grid graph. The underlying cellular units provide semantic information to the corresponding grid cells. In the grid graph, each grid cell has one and only one membership of a semantic, and the topological relationships among cellular units can be represented by the edges of the grid graph.

FIGURE 2.32 The principle of Wi-Fi indoor positioning. (a) Positioning from active points (from the webpage of Barcoding Inc. www.barcoding.com/wireless/asset_location_system.shtml); and (b) Fingerprinting map (from the webpage of Cisco www.cisco.com/en/US/docs/solutions/Enterprise/Mobility/wifich3.html).
Similarly, in robot motion, a planning occupancy grid approach uses a regular matrix of equally-sized cells for autonomous navigating robots [ME85, FMWN05]. In this matrix, each cell connects to its eight neighbouring cells (with the exception of boundary cells). A high probability value is assigned to grids in accessible/navigable spaces and a low value to grids occupied by objects.

A common Wi-Fi Positioning method is using the received signal strength indication (RSSI), i.e., the power level received by sensors [CNPM11, Mau12]. This method collects RSSI from all Wi-Fi active points (AP) (Figure 2.32a) in a building and generates a ‘fingerprinting’ map. The ‘fingerprinting’ represents the expected distribution of Received Signal Strength Indicator (RSSI) (Figure 2.32b), which provides a position estimation for a user who can measure the RSSI at a location and compare it with the ‘fingerprinting’ map [VZVW+13].

\[\text{FIGURE 2.33} \quad \text{The principle of IndoorAtlas. (a) Aligning a floor plan; and (b) The generated magnetic map (from: www.indooratlas.com/).}\]
Another similar method adopts the magnetic field for indoor positioning [Mau12, TMM⁺16], such as the application called IndoorAtlas [Ind16]. After aligning floor plan images with geo-coordinates (Figure 2.33a), IndoorAtlas collects data and generates a magnetic field map for the submitted floor plan (a type of fingerprinting map) (Figure 2.33b). As generally the magnetic field of each building is unique and different at various locations inside the building, this method can provide a relatively stable positioning result for the building.

With spatial information, one can conduct indoor tracking (i.e., continuously following the trajectory of someone) with different localization devices. Commonly indoor tracking methods include: Dead reckoning (DR), Grid filter, Map matching, and Model-based approaches [Mau12]. DR computes a person’s current location by advancing a known position with course, speed, time and distance to be travelled. DR data can be collected by inertial measurement unit (IMU) [Mil06, Mau12] on tracking devices. The uncertainty of dead reckoning positions grows with time thus it is necessary to check the position regularly [Mil06].

A grid filter is a kind of discrete Bayesian filter, which probabilistically estimates a target’s location based on observations from sensors [Mau12]. This type of method is widely used in the field of robotics [BFH97, TBF05]. They compute the location in two phases: the prediction phase where the prior probability of location is estimated based on the previous location, a motion model and the map of tracking environment; and the update phase where the posterior probability is computed by multiplying the prior probability with a conditional probability. The conditional probability is computed according to the measurements of sensors.

Map matching assumes a user can only be located along certain routes [Mau12]. Some constraints on indoor environments are applied to refine estimates of the moving positions of a person inside a building. For instance, a user does not pass through walls, but only along corridors and through doorways [Mil06]. Basically, there are two map-matching techniques: point-to-vertex matching (i.e., a measured location to a vertex in the route), and point-to-edge matching (i.e., a measured location to an edge in the route). An implementation of point-to-edge matching shows satisfied results in a corridor environment [Spa07].

Model-based methods adopt a vector model of an indoor environment to improve the estimation of user locations. This method can be taken as an extension of map matching methods. They consider model features (such as walls or obstacles) [GCZ⁺11], sensor information (e.g., speed and direction), and information from users (e.g., mean velocity and velocity variance [KKRA08]). Jensen et al. [JLY09] propose a base graph model for tracking which represents the connectivity and accessibility of indoor space.

There are some other tracking methods for users. Optical tracking can be regarded as an alternative for users when positioning systems cannot work smoothly, such as identifying a user’s location by matching photos provided by the user [ZVO4]. Surveillance videos can also be applied to indoor user tracking. Zhou et al. [ZZW⁺16] adopt surveillance videos to extract single pedestrian traces, and then match the depth information of 3D scenes to the created indoor navigation network. In this way, pedestrian traces are reconstructed and visualized in the 3D indoor environment. This method is suitable for building managers who need to monitor a specific person’s (e.g., maintenance staff) trajectory in this building.
Girard et al. [GCZ⁺11] propose a real-time indoor navigation solution without pre-installed sensor networks, which combines four existing techniques: foot-mounted Inertial Motion Unit, ultrasonic ranging, particle filtering and model-based navigation. This solution shows an accuracy improvement compared to the previous indoor localization methods.

By using the Quuppa positioning technology (a localization system using a unique Angle-of-Arrival method of Bluetooth Low Energy signals) [LLC16], Van der Ham et al. [vdHZVV16] track the location of hospital assets. In addition, indoor spaces are subdivided according to these assets to improve accuracy. With such a subdivision the test shows a relatively good result [vdHZVV16]: there are four results with expected accuracy out of six cases.

In addition, a spatial model-aided approach is proposed to improve indoor tracking results [Xu14, LXPZ15]. This method integrates geometrical, topological and semantic features of a building to exclude locations with lower probabilities and improve a user’s track from disordered measurement.

§ 2.6 Adopted methods in this thesis

For different components of indoor navigation, this research as described in the thesis selects and adopts specific methods concerning building models, navigation models and routing methods.

**Building models.** CAD models, IFC of BIM [IAI16] and CityGML LOD4 [GKNH12] data are all adopted in this thesis as input for indoor routing. The semantics of building are distinguished from the input data, according to a specifically designed data model (which will be introduced in Chapter 3). These semantics generally reflect navigational functions (e.g., stairs assigned semantics Vertical Unit to connect distinct floors) of indoor spaces applied to different applications. In addition, the semantics of the designed data model are readily created or converted from the mentioned building models.

**Navigation model (network).** I take navigation networks as a navigation model since I want to explore the combination of topological and geometric networks for routing. Specifically, the used topological network is a pure graph structure and the geometric network is tagged with geometric coordinates. The navigation models represent a specific hierarchical model which separates the topological and geometric details from building models. In this way, indoor routing becomes simpler and more effective with the two navigation models: routing on the topological network excludes spaces according to certain criteria; then accurate paths in the selected spaces are computed in the limited geometric network of these spaces. Additionally, the specific hierarchical model contains just two levels of networks to avoid maintaining complex relationships among multiple levels. In other words, this research does not consider a hierarchy of different types of space (e.g., section, room, and sub-room).

A Visibility Graph (VG) method [OW88, AW88] is used to support geometric network generation. An obstacle-avoiding path between two locations can be computed based
Routing methods. As this research focuses on the single-source path finding, the classic Dijkstra algorithm [Dij59] is adopted to find the shortest paths with different types of weights.

Routing criteria are designed for topological network, which is about the selection of spaces. The expected routing result is in the form of sequential rooms/spaces to be passed. These criteria are related to the centrality of a network and semantics of spaces (e.g., to minimize stairs in a path). In addition, three heuristics [CTG+06] for indoor wayfinding (central point strategy, direction strategy and floor strategy) are adopted to construct criteria for indoor routing (see Chapter 4). To reduce multiple candidate paths resulting from routing, Lexicographical Goal Programming [JT10] was adopted (see Chapter 4).

Routing based on geometric networks shows how to accurately navigate inside rooms / spaces, in terms of obstacle avoiding and accessibility to a given user. Thus the routing on a geometric network focuses on the accessible shortest path inside a space or between different spaces. The criterion of accessibility, which is related to user sizes, dominates the routing on geometric networks. In order to avoid and cross between obstacles, the minimum distance is used to indicate the ‘bottlenecks’ among obstacles. These bottlenecks among obstacles reflect a configuration of accessible regions in a space. Bottlenecks can be compared with a user size and then the accessible regions delimited for the user. The accessible regions can be computed not only for wheelchair users, but also for other applications (e.g., a person manipulating an indoor vehicle).

As mentioned in Chapter 1, indoor positioning techniques and indoor wayfinding is not researched in this thesis. Therefore, this thesis does not discuss or select related methods for the two purposes.

§ 2.7 Summary

This chapter introduces related research on different components of indoor navigation, i.e., building models, indoor navigation models, routing algorithms, human wayfinding behaviors and indoor positioning techniques. Based on this chapter, the following research question is considered:

1. What kind of information, data models and routing algorithms has been used and developed so far, and what are their limitations for large complex buildings?

In general, indoor routing needs the semantics, topology and geometry of buildings. Indoor routing requires building models (Section 2.1) as input, such as CAD files of floor plans, standard data of city models (i.e., CityGML) and standard data of buildings (i.e., BIM IFC). CAD files lack the semantics of indoor spaces and the geometry can be very primitive (e.g., lines). Semantic models of CityGML and BIM IFC contain abundant space semantics and accurate geometry of these spaces (e.g, 2D surfaces or 3D...
Another standard model named IndoorGML focuses on indoor spaces and their connectivity for navigation networks. The navigation networks have to be converted from the building models. Semantics of indoor spaces should also be considered in the generation of navigation networks. There are many ontologies which define semantics in different ways. In order to apply any of the ontologies, one needs to consider the method of subdividing the building (e.g., based on structure or functions of spaces). Because the subdivision method is not clear, semantics of the reported ontologies are either too general or too detailed for different cases.

Navigation models (i.e., 2D/3D networks or grids) can be generated from the building models for indoor routing (Section 2.2). In general, there is no standard navigation model for every case of indoor navigation. Therefore, a navigation model needs to be selected according to the specific context.

Although shortest path algorithms such as Dijkstra [Dij59] and A* [HNR68] are the base for indoor routing, many researchers propose ad-hoc routing methods and routing criteria relying on specific navigation models and user needs (Section 2.3). Except for distance, travel time or number of turns, non-metric factors are also used for routing (e.g., cognitive similarity, temperature, and visual signs). Some pedestrian-related paths are also defined, such as a ‘feasible’ and ‘comfortable’ path for a wheelchair user [DGK09a], Least-effort, or Least-visible paths [LZLF08, CWSC14]. Other routing methods are defined on specific navigation models or structures, such as cactus tree-based routing [WMY07], routing on an Octree structure from point clouds [RVZ16], etc. However, two issues are seldom discussed for indoor pedestrian routing: 1) routing according to space semantics; 2) dimension of pedestrians. This thesis considers and designs related approaches for both the issues. First, different criteria are proposed to support indoor routing with space semantics (Chapter 4); second, a new indoor routing approach is developed to consider the size of users (Chapter 5).

The pedestrian wayfinding research (Section 2.4) shows some heuristics regarding human wayfinding behaviours, which can be considered to design routing criteria. Although indoor positioning techniques (Section 2.5) are important to provide and trace a user’s location, in this thesis positioning is not the focus of indoor routing since the location can be reported or assigned by users.

Section 2.6 presents the methods adopted in this thesis, which is the starting point of this PhD research. This research adopted building data (CAD files, CityGML LoD4 and BIM IFC), navigation networks, and the Dijkstra algorithm for routing. Based on these existing methods, the focus is to facilitate the generation of navigation networks from building data, and on user needs of passing through specific locations and/or spaces in path computations. In addition, user size is considered for accessible path computation in this research. The next chapter will present the developed data model mapping indoor spaces to navigation networks according to space functionalities, and the two-level routing approach on these networks which considers user needs and their sizes.
3 Space modelling for two-level routing

This chapter presents an indoor routing approach which incorporates building information on two levels, i.e., abstract and detailed levels (which relate to logical and geometric networks, fully described in Chapter 4 and Chapter 5, respectively). The two levels are built based on indoor spaces. In this thesis an indoor space is defined as a volume with a conceptual or physical boundary. Such a space can be partially occupied (e.g., by the volume of a coffee machine or other furniture/obstacles) or empty. The two levels refer to two navigation networks regarding indoor spaces. The navigation network on the abstract level provides a topological representation of a building, e.g., a connectivity graph among all the spaces of the building. The navigation network on the detailed level represents accurate paths through geometric locations along various obstacles related to these spaces. The two-level routing integrates both the networks to compute paths considering user preferences for spaces and/or geometric locations.

Section 3.1 explains the basic terms used for the two-level routing approach and defines navigation networks on the two levels. In Section 3.2, a conceptual data model is proposed to capture different spaces of a building and depict the relationships of the spaces. This data model is used to bridge building data to navigation networks on the two levels. Section 3.3 elaborates the relationships between navigation networks on the two levels and the proposed data model. Section 3.4 presents routing options either on the abstract or the detailed level and introduces some combinations that use the two levels together for routing. Compared to a one-run routing in a whole navigation network, the two-level routing approach provides more flexibility (e.g., to get an abstract and/or a detailed path according to user demands) and more routing functionalities (e.g., through assigned ordered spaces of interest (SOI) and points of interest (POI)). This chapter is related to the following author’s own publications: [LZ12, LZ13b].

§ 3.1 Concept, definitions and terminology

This two-level routing concept is the theoretical foundation for the whole thesis. The two-level routing approach is a synthesis computation integrating routing on both the abstract and detailed levels. On the abstract level one can compute and adapt a conceptual path (i.e., a sequence of spaces) to user needs, on the detailed level one can obtain accurate paths that suit user sizes to avoid obstacles. Routing on the abstract level enables a user to add her/his preferences for specific spaces. Depending on spaces in a computed conceptual path, a navigation network on the detailed level is built in these spaces with consideration of user size. In other words, the two-level routing is not based on a pre-stored complete navigation network that covers all the spaces of a building. Instead, the indoor navigation networks are generated ‘on the fly’, and they are easily recreated when the indoor environment partially changes (e.g., a space is locked or furniture is moved). In the two-level routing approach, an ‘optimal’ (e.g., the shortest distance or time) path is not the focus. Because this approach gives priority to indoor navigable spaces and their functions for routing, and takes the spaces to delimit indoor paths by determining different space sequences according to user profiles.
FIGURE 3.1 Concepts of indoor space. (a) Description of Space, Navigable Space and Non-navigable Space; (b) The dual graph of Space 1 and Space 2. The graph is bi-directional; and (c) POI, Geometric node and path.

and preferences. Geometric paths (on the detailed level) are refined in a given space sequence (e.g., a path on the abstract level). This coarse-to-fine approach of two-level routing could exclude a part of the indoor spaces on the abstract level, and thus the so-called ‘optimal’ paths are not ensured.

The introduction of all concepts starts from indoor space. An indoor space (‘space’ for short) is a volume physically or virtually bounded. A room in a building is considered space. The connection volume between two rooms is named an opening, and the volume of opening is also space. An opening can be occupied by a door/window panel. Indoor objects are regarded as static obstacles. They are also space that may occupy a part of a room. Thus a room can be fully empty, partially occupied (e.g., a room with obstacles) or fully occupied (e.g., that of an obstacle). It is possible to have one space in another (pillar in a room, see Figure 3.1a). For a room with static obstacles, the free part is named Navigable Space for motions, while the obstacle part is named Non-navigable Space. A Wall is another type of Non-navigable Space (Figure 3.1a). It bounds a room from the outside and no human or robot user can move in it.
Chapter 2 (see Section 2.1) has introduced structural and functional subdivision of buildings, which can result in physical and virtual spaces, respectively. These spaces can be further split into more small pieces or integrated into a larger one. A building is finally represented by a number of spaces. A dual graph of these spaces can be created: selected spaces are abstracted as nodes, and edges of the graph represent connectivity (see Figure 3.1b). This dual graph is the navigation network on the abstract level. Space semantics can be assigned to related nodes. As these edges just represent connectivity, no semantics would be assigned to these edges. But edges can indicate the direction of access permission of a space and motion modes of humans (e.g., edges for a walking person or wheelchair user).

Two important concepts are introduced in this thesis: Space of Interest (SOI) and Point of Interest (POI, see Figure 3.1c). A SOI is a region which a user wants to visit. Examples of SOIs are coffee machine neighbouring areas, registration desk front areas, waiting areas, a specific door or even a user-defined place. In particular, a SOI is a whole (i.e., a corridor) or a part of space.

SOI refers to the space as a concept (i.e., the name of a room or a place), POI is a location at or close to a SOI, or contained in a SOI (when the SOI is a whole space). A POI is given with its three-dimensional coordinates, e.g., a point in front of the coffee machine. Generally, a POI can be any location in a building. But a POI should refer to useful information, thus in this thesis a POI is defined as an abstraction of a SOI (e.g., functional spaces and physical spaces such as an indoor obstacle). For example, in the case of a door or window, POIs can be the centre point of the shape. A POI can also be created with indoor obstacles: a reference point of an obstacle is considered a POI. Normally the POI is close to the obstacle and accessible to the user, such as a nearby location to a coffee machine. Subspace would be created for the POI by users. The POI inherits the name of the related subspace (‘coffee machine’, ‘reception desk’, etc.). In this respect SOIs are used on the abstract level and POI on the detailed level.

This thesis makes a distinction between two categories of indoor navigation networks, i.e., logical and geometric (on the aforementioned ‘abstract’ and ‘detailed’ levels, respectively, see Chapter 1). The logical network is about the connectivity of indoor spaces and it represents an abstract routing network for a building, while the geometric network is created in navigable spaces, and it is the detailed routing network for the building. The two types of network are defined as a set of nodes and edges. Their definitions are presented as follows.

**Definition 1 Logical network.** A logical network is a directed graph $G_l = \{V_l, E_l\}$ where $V_l$ is the node set representing indoor spaces, and

$$E_l = \{e = (n_i, n_j) | (n_i, n_j \in V_l) \cap (n_i, n_j \text{ are the spaces sharing openings})\}$$

An edge indicates two spaces connected via an opening. In other words, edges of the logical network are connectivity of spaces and they are directional. Given a set of indoor spaces, the logical network is a connectivity graph regarding these spaces.

**Definition 2 Geometric network.** It is a network $G_g = \{V_g, E_g\}$ where $V_g$ is the node set including opening centres and Points of Interest (POI) in a set of indoor spaces, and

$$E_g = \{e = (n_i, n_j) | (n_i, n_j \in V_g) \cap (n_i, n_j \text{ are in the same space})\}$$
The nodes in $V_g$ represent indoor transfer locations (e.g., door centres) and POIs with coordinates. Normally nodes regarding openings are constant but POI nodes can be added on demand. The edges represent the obstacle-avoiding path with the shortest distance (the *shortest path* in short) between two nodes related to the same space. This definition of edge delimits the number of edges. Thus, in a geometric network, there is no edge for two nodes pertaining to different spaces. Then the shortest path between the two nodes can be computed based on edges. Edges of a geometric network are derived with the building geometry (i.e., shapes of spaces, obstacles and openings). An edge may not be a straight line but a polyline (with many intermediate points except the two nodes). A geometric network can be assigned to any number of spaces in a building. Given a set of indoor spaces, each space contains a subnetwork of the geometric network.

**FIGURE 3.2** Logical network of spaces. (a) Indoor spaces of a design plan; and (b) The logical network.

The term *logical network* refers to topological relationships only of indoor spaces regardless of geometric information of buildings. In this thesis, the nodes represent a building’s navigable spaces (e.g., rooms or corridors) without coordinates, i.e., they are not geometrically defined. The edges represent only the topological relationship (connectivity) between the spaces. The nodes inherit only the semantics of related spaces. Figure 3.2 provides the logical network for several rooms. No geometric information is attached to the logical nodes and the edges. The paths in logical networks are named *logical paths*.

Two geometric networks in Figure 3.3 are created with the same spaces as Figure 3.2 but with different geometric information. Figure 3.3a presents an example of visibility graph (VG); and Figure 3.3b presents the case of a straight medial axis. Paths in geometric networks are named *geometric paths* that consist of a sequence of nodes and related edges. As mentioned in Chapter 2, this thesis adopts a VG approach [AW88, dBcK008] to derive geometric networks.

The ‘two-level’ notion is inspired by an important A* technique named *Hierarchical Pathing* in computer game programming [Rab00]. In *Hierarchical Pathing*, the first
step is to find the overall path, and the second step is to refine path on the local level. Both the paths refer to geometric paths.

In this research, the notion of two levels is defined by using the logical and geometric networks. The abstract level focuses on connectivity of navigable spaces, and routing is conducted in a logical network of a building. The abstract level is used to indicate how to pass spaces. The detailed level focuses on the geometric aspect of paths. A geometric network is created on the detailed level within the selected spaces.

On the abstract level, routing relies on the spaces’ semantics, which will be elaborated in Chapter 4. In the next section, a data model for the representation of the semantics is introduced. The model contains dedicated semantics to support the generation of logical networks and to categorize spaces with the semantics.

Figure 3.4 gives a simple example of the logical and geometric networks of a single floor (Figure 3.4a). Consistent with the structural subdivision of the floor, the resulting navigable spaces are selected to form a logical network (Figure 3.4b). A logical path is defined on the navigable spaces R1, R2 and R3, and the geometric network (see Figure 3.4c) considering doors is created within these three spaces for routing (i.e., formed by three sub-networks of each space). The constructed geometric network is based on the shortest paths between the doors (Figure 3.4d). Such networks and related routing on the detailed level will be elaborated in Chapter 5.
FIGURE 3.4 Illustration of logical and geometric networks. The logical network is the ‘space-to-space’ style. (a) A floor of five spaces; (b) A logical network for spaces on the floor; (c) A related geometric network. Nodes are door centres, and edges represent the shortest paths among the nodes; and (d) The geometric network only for R1, R2, and R3.
§ 3.2 Indoor navigation space model (INSM)

Indoor space is a fundamental aspect for indoor navigation. These spaces derived from a whole building space with given semantics contribute to indoor routing, because indoor routing is based on the navigation models of these spaces. As addressed in Chapter 2, shortcomings of previous research on pedestrian navigation networks can be summarized by the following:

1. A limited link between indoor space subdivision strategies and routing. The generation of navigation networks for different buildings is not well defined. Most research shows the navigation networks of simple-structured buildings, while few studies discuss those of complex buildings where different subdivision strategies can be applied.

2. Separate semantics of indoor spaces, or even no semantics in some applications. On the one hand, some navigation networks adopt different semantics which have different names for the same type of space or provide distinct definitions of spaces and objects (e.g., the models IndoorGML and 3DBO, see Chapter 2). On the other hand, many navigation networks are purely geometric without space semantics.

3. Limited consideration of indoor routing with obstacles. A few navigation networks [GZ11b, LYJS09, SR09, MZLC14] link obstacles to the network generation and routing. But the influence of changes on obstacles or users is not clearly addressed. In contrast, obstacles and user sizes are commonly considered in robot navigation whose routing is mostly based on Minkowski sums [dBCvKO08, Coe12] (see Chapter 2).

This section presents a proposed spatial-semantic coherent data model named the Indoor Navigation Space Model (INSM). This model is specifically designed to define indoor spaces by their navigational functionalities, and to support indoor routing in different environments/scenarios with distinct subdivision results. The INSM model concentrates upon the ‘functional’ spatial semantics such as distinct spaces specifically for horizontal and vertical motions (corridors and stairs), and the connection part of these different spaces.

Moreover, the INSM is developed to distinguish the use of semantics and geometry of buildings. As mentioned above, semantics of different data models (e.g., CityGML and BIM IFC) are not completely compatible and they are seldom directly used for indoor routing. Thus, INSM semantics are designed to support indoor routing where semantic routing criteria can be developed (Chapter 4).

In terms of indoor semantic models, there are two prominent alternatives for routing i.e., 3DBO and IndoorGML (see Chapter 2). The semantics of 3DBO are too specific and detailed; it separates similar spaces (e.g., passage and corridor), which is unnecessary according to space functionality (passages and corridors are both for horizontal movements).

The other data model - IndoorGML - is more general and similar to INSM. The main difference between them is that IndoorGML only contains a network (the Core module). The network of navigable spaces (the Navigation module) is not compulsory for IndoorGML. IndoorGML defines the network of indoor spaces based on space connectivity, which can be used for the logical network (i.e., the abstracted level), but more
geometric information is not included in the IndoorGML. For example, POI is not explicitly defined in IndoorGML, and POI can frequently be used for routing. In contrast, INSM is space-centred and facilitates the generation of navigation networks of the two-level routing. In fact, INSM is proposed earlier than the IndoorGML and it is specifically designed to support the two-level routing.

In general, INSM is devised to manage functional space semantics, topology and geometry. INSM regards each component of a building as a space (either occupied by objects or not), and the semantics of a space indicate its functionality in a navigation process. As INSM explicitly contains connectivity among spaces, logical networks can easily be generated. Furthermore, with INSM the semantics of spaces can be propagated to nodes of logical networks. The building geometry stored in INSM can be used to create specific geometric networks, which derive obstacle-avoidance paths for users with different sizes.

The main characteristics of INSM are:

1. Space subdivision of a building into non-overlapping spaces;
2. Dedicated semantics which represents these spaces according to their functionalities for indoor routing;
3. Possibility to automatically derive a semantically rich logical network. The generation process uses the concept of Duality as in IndoorGML (see Section 2.2);
4. Possibility to automatically derive geometric network.

Indoor spaces referred by INSM can be either three-dimensional (3D) volume/solid or two-dimensional (2D) surface/area. For simplicity, in the implementation of these concepts (see Chapter 6) paths are computed based on the non-overlapped surfaces in 2D and 3D spaces (e.g., 2D for horizontal spaces and 3D for stairs). Therefore, the INSM discussed below adopts surface geometry to represent spaces.

Following the data modelling tool Sparx Systems Enterprise Architect, INSM is presented by two UML class diagrams which are independent of platform- and technical-specific information. The first one contains only classes without attributes; the second includes the classes and their attributes.

The INSM model in the first form is profiled in Unified Modelling Language (UML) by using the Enterprise Architect (Figure 3.5). It includes the classes of indoor spaces and their aggregation classes, their compositions and other associations. Figure 3.6 presents the main part of the INSM model with attributes.

The fundamental classes to the INSM are the Opening (OPN), NavigableUnit (NU), and NonNavigableUnit (NonNU) (the yellow boxes in Figure 3.6). The following part elaborates on the main classes of the INSM.

**Definition 3 Opening (OPN).** This is a transition space which connects one space with another. These spaces (e.g., an entrance) can connect the outdoor space as well.

An OPN can be a Door, a Doorway, MainEntry or a Window. The OPN contains attributes that characterize properties of importance for navigation such as: the entity of OPNs...
FIGURE 3.5  The INSM model represented only by classes. All classes refer to spaces except PointOfInterest. PointOfInterest represents POIs (locations with coordinates). A part of the classes is aggregation classes, i.e., VerticalSpace, HorizontalSpace, NavigableBuildingSpace, BuildingPart and Building. The other classes refer to independent spaces.

(IsExisted), the adjacent spaces or spaces that the opening links (the association Space1 / Space2 in Figure 3.6), and accessibility of the space (IsLocked). The OPN has five sub-types, i.e., Door, FacadeWindow, InteriorWindow, Doorway and MainEntry. The type Door is critical for routing as it provides the connectivity of spaces. The type InteriorWindow and FacadeWindow can be used in routing in special cases (e.g., an emergency). FacadeWindow refers to the link to the outdoor space. The Doorway is a special type to depict cases where no physical boundary between two spaces exists. For example, the gap between two spaces, when a movable wall is removed, is a doorway. Lastly, MainEntry represents the connection to the outdoor space (e.g., the main entrance).

Definition 4 NavigableUnit (NU). This is a space in which users (e.g., pedestrians) can move freely (e.g., walk or drive) without crossing any opening.

Each NU class has as attributes the heights of the space (BottomHeight and TopHeight) and the Name of the NU. The attribute Name is used for the abstract routing, i.e., these are the names used by a user.

Definition 5 SpaceOfInterest (SOI). This is a sub-region of NU assigned by a user for specific purposes.

A SOI refers to a navigable region that can be contained in a NU or just the NU itself. The SOI reflects a user’s interest in visiting this specific region. For example, in a public space the front area of a coffee machine is a SOI, and a corridor (also a NU) is a SOI. Each SOI associates with only one NU, while a NU corresponds to none or many SOIs (see Figure 3.6), which indicates a user can separate several regions in a NU as SOIs. The class SOI includes the attribute Name to store the descriptor of the region.
**Definition 6 PointOfInterest (POI).** A POI is the reference location of a SOI.

The class POI is associated to the class SOI with a 1-to-1 relationship (see Figure 3.6). For instance, a POI can be door centres, obstacle corners, or any user-defined location in NUs. The class POI contributes to routing on geometric networks. More details on geometric networks will be given in Section 3.3. A NU may contain none or many POIs, but a POI can be only contained in one NU.

**Definition 7 NonNavigableUnit (NonNU).** This is a space occupied by objects in which pedestrians cannot be present.

The NonNU consists of two subclasses, i.e., the classes Wall and Obstacle (see Figure 3.5). An Obstacle is the inaccessible space inside NU, such as a pillar or furniture. In contrast, a Wall cannot be in a NU. A Wall is an inaccessible space that is adjacent to a NU and bounds the NU. In the 2D space, the representation of a NonNU is a surface and in 3D it is a volume. The association Space of the class Obstacle gives the NU containing the Obstacle. The class Wall can touch two NUs or one NU and the outdoors, which is given by its association Space.

The class NU represents all kinds of indoor spaces. Generally, two NUs are connected via one or more OPNs. The multiplicity between NU and the OPN is 1 to many, because a NU can relate to from 1 to multiple OPNs (e.g., doors), and conversely an OPN is associated with 2 NUs. In addition, the class NU is associated to the NonNU. The multiplicity between NU and NonNU is 1 to many, which indicates a NonNU may be related to 1 or many NUs. Conversely, a NU associates with many NonNUs.
Definition 8 VerticalUnit (VU). VU is a subclass of NU in which pedestrians can move (or be transported) in vertical directions (i.e., up and down) along the same slope.

Definition 9 HorizontalUnit (HU). This is a subclass of NU in which pedestrians can move in horizontal directions.

The Vertical Unit (VU) and Horizontal Unit (HU) (see Figure 3.5 and Figure 3.7) are exclusive subclasses of the NU. A user’s vertical and horizontal movements in a building are bounded to the two classes.

The VU has the association Top/Bottom to give the two HUs connected by it. Another attribute IsContainedVerticalUnit records whether the VU is contained in a HU. The HU indicates its HorizontalSpace (the definition will be given after) information by its aggregation HorizontalSpace (see Figure 3.6). Generally, an HorizontalSpace is similar to the general notion ‘floor’. A VU can connect two HUs or even more, such as an elevator connects every floor of a building. An HU may not associate any VU, or link many VUs (e.g., an entrance hall to several stairs).

Both the VU and the HU (see Figure 3.7) are too general to describe the use of a space for indoor routing. In fact, a user does not consider two HU spaces (e.g., a corridor and a balcony) equally important for routing. Similarly, VU spaces such as stairs and elevators are not equally important for routing. Therefore, subtypes are designed for the VU and HU.
The VU has three subtypes \textit{Stair}, \textit{Escalator}, and \textit{Elevator} (see Figure 3.5 and Figure 3.7). They support different modes of vertical motions in a building. The \textit{Stairs} are used for walking; a user can save effort by using the \textit{Escalators} and \textit{Elevator}. The three subtypes are simplified in the attribute Type of the VU (see Figure 3.6). The HU has three subtypes \textit{End}, \textit{HorizontalConnector (HC)} and \textit{VerticalConnector (VC)} (see Figure 3.5), which are introduced below.

\textbf{Definition 10 End.} \textit{End} is a subtype of HU which is connected with one NU at most.

\textbf{Definition 11 VerticalConnector (VC).} \( VC := \{ NU_i \in NU \mid \forall NU_i, \exists connected \ NU_m, \ NU_n \in NU, m \neq n : (NU_i \in HU) \cap (NU_m \in VU) \cap (NU_n \in VU) \}. \)

**FIGURE 3.8** An example of the VC. The iron platform is a VC connecting the two escalators.

A VC is a HU which connects at least two other different NUs, and at least one of them is a VU. Although a VC is for horizontal motions, its name include ‘vertical’. That is because the VC connects a VU at least, and it is the joint connecting the horizontal and vertical parts. In the case that a HU directly connects to a VU, a VC may be a virtual space that includes no physical walls. A VC (see Figure 3.8) can be divided from the HU manually. The VC benefits the routing with semantics of spaces: VCs would be searched first when a user needs to switch floors. The VC is an indication of possible floor changes. In contrast, an HC, another subtype of HU, cannot provide such information.

The VC is an important subtype in the INSM. Although a HC and a VU may connect each other directly, the VC is an indication to find VU quickly. It is called VC because it connects vertical parts. A VC is a virtual space in a space which is contained in a HU. The semantics of the VC refer to connections of vertical and horizontal parts, which has benefits for routing.

A VC is supposed to be a small space covering the entrances of the VU, and therefore it bridges the VU and the other HU. Thus a VC cannot be a HC (connecting at least two HUs) at the same time. The size of a VC is flexible, which can be decided according to its capacity of pedestrians [KZ14]. In Figure 3.7, the VC connects the two stairs to the upper and lower floors, respectively. Meanwhile, the two stairs are connected to a HC via this VC.

According to the VC’s definition, the subtype VC is associated to the VU and HU. A VC may connect to 1 or many VU, and link to 1 or many HU. As a VU bridges at least two floors, the VU connects at least two VCs. A HU may be not related to a VC (i.e., only to other HUs), thus a HU is associated to none or many VCs.
A HC is a HU which connects at least two other different HUs, and all of them belong to the same HorizontalSpace (see Figure 3.7). Thus, a HC associates to at least two other HUs (see Figure 3.5). Conversely, a HU may be an End, then a HU connects to none or many HCs (see Figure 3.5 and Figure 3.7). The definition of HC will be provided after the definition of HorizontalSpace.

Definition 12 HorizontalSpace (HS). A HS is a collection of NUs, where the maximum of their top heights (topH) is denoted by MaxH, and the minimum of their bottom heights (botmH) by MinH. These NUs include both HUs and VUs whose botmH and topH are not lower than MinH and not higher than MaxH. \( HS := \{HU_i \in HU, VU_k \in VU | \forall HU_i, \exists (MinH \leq botmH_i < MaxH) \cap (MinH < topH_i \leq MaxH); \forall VU_k, \exists (VU_k \text{ is contained in } HU_i) \} \).

The class HS is an aggregation class of the two classes HU and VU. The two values MaxH and MinH are stored in the attributes MaxHeight and MinHeight of HS (see Figure 3.6). A complicated floor that contains intermediate levels can be represented by the HS. In the INSM model with classes and attributes, HU aggregates to HS in the relationship HorizontalSpace and VU uses an aggregation IsContained to indicate the VUs contained in HS. A HS includes one HU at least (the multiplicity 1...*). The HS may contain none or many VUs, thus in the IsContained aggregation relationship the multiplicity on the VU side is from 0 to many. Figure 3.9 presents the case that a HS contains some steps as a VU. Here the contained VU is useful to depict irregular shapes inside a building, especially for the case that a floor contains small steps or a small stair to a hanging platform (see Figure 3.9).

Definition 13 HorizontalConnector (HC). A HC is a HU that connects with at least two other HUs. \( HC := \{HU_i \in HU | \forall HU_i, \exists HU_m, HU_n \in HU, HU_i \in HS : (HU_i \text{ connects } HU_m) \cap (HU_i \text{ connects } HU_n) \cap (HU_i, HU_m, HU_n \in HS) \} \).

The definition of the class HC relies on the HS, because a HC connects with other HUs on the same HS. The HC represents passages, corridors and halls in a HS.

Definition 14 VerticalSpace (VS). A VS is a group of VUs whose maximum height difference is less than a threshold MaxDist. \( VS := \{VU_i \in VU | \forall VU_i \notin CVU, \exists VU_j \in VU \text{ and } VU_j \notin CVU : \text{ the distance between } VU_i \text{ and } VU_j < \text{MaxDist} \} \).
FIGURE 3.10 The VerticalSpace (in blue) and HorizontalSpace (in green). There are four HorizontalSpaces (e.g., a part of or a whole floor) and two VerticalSpaces (e.g., stairs).

The VS is the aggregation class of VU. A VU is either aggregated to the VS, or to the HS (Figure 3.6). A VS contains at least one VU, then the multiplicity on the VU side is from 1 to many. A VS represents a complete staircase or elevator, and a VU is only a part of the VS. The class VS has an association Top/Bottom which refers to the connected bottom and top HS of the VS. A VS connects to at least two HSs (multiplicity from 2 to many on the HS side in Figure 3.6). In contrast, the multiplicity is from 0 to many on the VS side in the association Top/Bottom. This is because a HS may not connect to any VS. Figure 3.10 presents examples of the VS and the HS. The green parts are HSs, and the blue parts are VSs. The VSs have different ranges, which means they connect to different top or bottom HSs. For the two HSs on the top level, one of them connects to a VS, and the other one is not related to any VS. The remainder of the HSs links to the two VSs.

Definition 15 NavigableBuildingSpace (NBS). The NBS is the collection of all the NUs of a building. \( BS := (VS \cup HS) \).

The NBS is an aggregated class which represents part of or the overall navigable space of a building, such as all the walkable spaces in two HSs (e.g., floors) and in the related VSs (e.g., stairs) connecting the two HSs. In addition, the NBS may not include VSs, and it has a HS at least, such as a construction with only one floor. The NBS class contains an attribute Name to store its depictor.

Definition 16 BuildingPart (BP). The BP is the collection of all the NUs, NonNUs and OPNs of a building. \( BP := (NBS \cup OPN \cup NonNU) \).

The BP is aggregated by NBSs, OPNs and NonNUs. A BP contains 1 or many NBSs, 1 or many OPNs and 1 or many NonNUs (see Figure 3.6). The NBS refers to the ‘free’ space in a building. Together with OPNs and NonNUs (e.g., walls), the BP refers to a complete notion of building space where the NBSs are connected with OPNs but occluded by NonNUs. The BP class has the attribute Name.

Definition 17 Building (BLD). The BLD is the aggregation of BPs. \( BLD := (BP_1 \cup ... \cup BP_k, k \geq 1) \).

An example of the Building (BLD) is two constructions connected by a connecting bridge. The two constructions and the bridge are BPs. The BLD class includes an attribute Name to store the name of a building (see Figure 3.6). The BLD uses Address to record the...
unique address of the building. A BLD may consist of one or many BPs (e.g., constructions connected by bridges).

In this thesis INSM is compared to two semantic data models, i.e., IndoorGML [LLZ+14] and the 3D Building Ontology (3DBO) [GZ11a]. IndoorGML is a standard of Open Geospatial Consortium, and it is worthwhile to clarify the connection between INSM and IndoorGML, which can benefit data transformation from the standard dataset. Semantics of 3DBO are comprehensive but no related navigation cases are reported. With the comparison to 3DBO, one can find in INSM the more concise semantics necessary for indoor routing. For example, Room and Hall of 3DBO can both be mapped to HU, and Corridor and Horizontal Passage of 3DBO can be sorted to HC (see Figure 3.12).

Compared to IndoorGML, the semantics of INSM is proposed earlier (in 2012) and more specific for navigation. Figure 3.11 presents the relationships between the two sets of semantics. The semantics of HC, VC and VU in the INSM are all equivalent to the TransitionSpace of IndoorGML (see Figure 3.11). Thus, horizontal and vertical spaces referred to by the HC, VC and VU are not separated in the IndoorGML as well. The navigation module of IndoorGML separates vertical and horizontal spaces according to the types of space provided by the OmniClass [Sec16] standard. For example, a TransitionSpace, can be regarded as ‘horizontal transition’ using code 1000 and as ‘vertical transition’ using code 1010 [LLZ+14]. However, this coding is not explicitly visible in the logical network.

**FIGURE 3.11** The associations of essential semantics of the INSM and the IndoorGML.

Detailed indoor semantics certainly bring more information for semantic paths. The 3DBO provides more details of spaces than the INSM (see Figure 3.12). The semantics of the Horizontal Passage, Room, Hall and Corridor from 3DBO associate with the HU and its subtype HC of the INSM. The 3DBO supports paths with more specific semantics on horizontal spaces, such as the Room, Hall and Corridor, which have no counterparts.
within the INSM. However, I argue that INSM presents a more concise and efficient perspective of space semantics. As mentioned above, examples are Hall and Room to HU, and Corridor and Horizontal Passage to HC (see Figure 3.12). Too detailed distinction on spaces (e.g., in 3DBO) would not necessarily promote indoor routing. In such a case, INSM just focuses on the navigational functionality of space (i.e., horizontal or vertical, Connector or End).
§ 3.3 Logical and geometric networks based on INSM

As mentioned above, the INSM is designed to facilitate the extraction of logical and geometric networks from geometric building models. Two INSM classes, the NU and OPN, are related to nodes and edges of logical networks. The classes NU and OPN are also used to derive nodes of a geometric network, but in a different way. Besides network generation, the INSM is used to semantically enrich the logical network. This section elaborates on the way that the logical/geometric networks can be derived from INSM.

The classes Node and Edge with the stereotype LogicalNetwork represent nodes and edges of a logical network (see Figure 3.13). Within the logical network, each NavigableUnit (i.e., NU) is represented by a Node. The class Edge represents the connectivity between NUs (Figure 3.13). One Edge refers to one or more Opening (i.e., OPN). For example, two spaces are connected via three doors. This connectivity is reflected by an Edge of the logical network. Note, no matter how many doors are available between two spaces, the connectivity is always given by one edge. Thus, a logical network is constructed by extracting each NU with a node and each connectivity relationship via the connecting OPNs between two NUs.

Figure 3.13 shows that Nodes can be specified with different INSM semantics (e.g., VU and HU). In this way, all Nodes can be tagged with the INSM semantics and then the logical network is the semantically enriched network.

Figure 3.14 presents the data model of the INSM and the logical network together with their class attributes. Subtypes (see the data type SubtypeOfNavigableUnit) of NU are stored in the attribute Type. Similarly, the classes OPN and NonNU store their subtypes in the attribute Type. The class Edge associates the class Node with the relationship Start/End (see Figure 3.14). Another attribute Type of the Edge is about the connectivity type of Edge, i.e., for indoor connections or between indoor and outdoor spaces.
The class Node has an association to the related NU. Each Node instance corresponds to one NU.

The classes NU and OPN also correspond to nodes and edges of a geometric network. Figure 3.16 presents the UML model of a geometric network with attributes of the classes. Four classes (i.e., the light green part) are about the geometric network, i.e., GeometricNode, GeometricEdge, OpeningNode, and PointOfInterest. GeometricNodes and GeometricEdges form a geometric network. An instance of GeometricNode is represented by a point with coordinates of a location, while instances of GeometricEdge are indicated by a polyline representing the shortest path between two GeometricNodes. GeometricNode has two subclasses, i.e., OpeningNode, and PointOfInterest. The classes OpeningNode and PointOfInterest refer to nodes of a geometric network. PointOfInterests are specified to functional spaces derived from a subdivision [KZ14], such as the front area of a coffee machine in a hall. A user can specify PointOfInterests to any indoor location, even for doors and windows. To get a geometric path, a user also needs to specify two OpeningNodes/PointOfInterests, respectively, as her/his start and target locations.

An OPN associates with one OpeningNode, which means the OpeningNode represents the OPN’s reference location (e.g., the centre). A GeometricNode is an accessible location with coordinates, such as door/doorway centres and other POIs. A user can specify a preferred indoor location as a GeometricNode (i.e., POI). A PointOfInterest associates a SpaceOfInterest with a 1-to-1 relationship. In the association between the class NU and the class SpaceOfInterest, the multiplicity on the SpaceOfInterest side is from 0 to multiple (see Figure 3.15), which means the implicit association between NU and PointOfInterest has the same multiplicity. Thus, a NU may contain none or many PointOfInterests. In contrast, a SpaceOfInterest belongs to only one NU, which implies a PointOfInterest also belongs to only one NU.
FIGURE 3.15 The UML model on relationships between INSM and the geometric network. The green part represents node classes (GeometricNode, PointOfInterest and OpeningNode) and edge class (GeometricEdge) of the geometric network. The pale pink part is INSM classes.

In the two-level approach, the geometric network does not have to be permanent. Different geometric networks can be derived in one building on demand. A geometric network can cover only one space, a group of spaces or all the spaces of a building. The GeometricEdge represents an accessible path between two GeometricNodes in the same space. Depending on the subtype of GeometricNodes (OpeningNode or PointOfInterest), a GeometricEdge may be from a door to another door, a POI to a door, a POI to another POI, and so on. In the association of the NU and the GeometricEdge, an NU contains 1 or multiple GeometricEdges. Conversely, a GeometricEdge belongs to only one NU.

A user in a NU needs to avoid OBSs (such as desks, chairs, etc.). A user’s motion is restricted by her/his size. Given a user size, the space between some OBSs are not accessible and the OBSs need to be put in a group. The user needs to avoid the boundary of the group of OBSs. The self-association of the class NonNU refers to the obstacle-grouping operation (Figure 3.16). One OBS may be grouped with none or many other OBSs.

On the one hand, a number of OBSs (e.g., desks) can be grouped into a larger one due to the user size; on the other hand, a NU can be subdivided into smaller ones to better represent the functional meaning of the spaces. In both cases, a GeometricEdge reflects the accessible path avoiding the OBSs between two GeometricNodes of the same NU. A GeometricEdge is represented by a polyline. The GeometricEdge (i.e., the accessible path) between two GeometricNodes may be different for distinct user sizes.

Figure 3.16 presents the data model of a geometric network which contains classes’ attributes. The geometric network is implemented by three classes (in light green), i.e., GeometricEdge, OpeningNode, and PointOfInterest. The GeometricEdge’s association Space refers to the space containing the GeometricEdge. The Geometry attribute stores...
FIGURE 3.16 Data model with class attributes on relationships between INSM and the geometric network. The yellow boxes are the core INSM classes (the NavigableUnit, Opening and NonNavigableUnit); the pale pink boxes represent three data types; the green boxes are classes of nodes and edges of the geometric network.

A GeometricEdge as a polyline. Another attribute Length records the distance of a GeometricEdge.

For the association between the classes GeometricEdge and the GeometricNode, a GeometricEdge connects two GeometricNodes, and conversely a GeometricNode may connect 1 or more GeometricEdges (the multiplicity 1...* on the GeometricEdge side) (Figure 3.16). Two of GeometricEdge’s attributes StartNodeType and TargetNodeType indicate the type (OpeningNode or PointOfInterest) of a GeometricEdge-related node. Four combinations of the start node and the target node are given: 1) from an OpeningNode to a PointOfInterest; 2) from a PointOfInterest to an OpeningNode; 3) from an OpeningNode to an OpeningNode; and 4) from a PointOfInterest to a PointOfInterest.

The class GeometricNode has two subclasses OpeningNode and PointOfInterest, and its attribute Geometry contains the coordinates (see Figure 3.16). The OpeningNode has the attribute IsPoI which indicates whether the OpeningNode is a point of interest to a user for routing. As mentioned above, each PointOfInterest belongs to just one NU, reversely a NU may contain none or many PointOfInterests.

A UML class diagram is used to arrange the classes of INSM, the logical network, and the geometric network in Figure 3.17. The key classes of the INSM are the NU and OPN. As mentioned before, the NU and OPN are associated to the classes that represent nodes and edges of the logical (in dark green) and geometric (in light green) networks.

Figure 3.17 presents the relationships among classes of the logical and the geometric networks. Nodes of the logical network refer to NU instances, and edges represent the connectivity of NUs. Nodes of the geometric network correspond to locations. An Edge
instance represents the connection of two NUs, which involves one or more Opening instances and thus corresponds to one or more OpeningNodes due to the 1-to-1 association between Opening and OpeningNode (Figure 3.17). A logical node (the class Node) contains none or more PointOfInterest instances, since each Node associates with one NU. The class GeometricEdge refers to a detailed path represented by a polyline and contained in a NU.

Generally, the logical and the geometric network focus on different levels of details; they are two ways of abstraction for one building. The logical network contains the semantics and connectivity of a building, while the geometric network is derived from the geometry of a building. Nodes of the logical and the geometric network refer to different spaces (e.g., NUs and Doors), thus logical nodes cannot be reused by the geometric network. Specifically, logical nodes are conceptual and geometric nodes are physical.

§ 3.4 Routing options

The previous sections have introduced INSM and its relationships with the logical and geometric networks. Both logical and geometric networks are used to provide routing. This section explains how they can be used individually or together. As will be shown later, one logical path can link to different geometric paths. Normally indoor path com-
putation provides one ‘optimal’ geometric path, but here geometric paths are computed on demand according to different conditions.

The combination of logical and geometric networks is very flexible and allows a variety of user-specific paths to be computed. Some users may be satisfied with a rough description of the path, which can be provided by using only the logical network. For instance, if a user only needs to know the names of the spaces to be passed (e.g., corridor and stairs) for orientation, then the path is represented by a logical path. If a user needs a more detailed path which shows how obstacles have to be avoided and which doors have to be used, then a geometric path is computed. Moreover, the two-level routing approach computes both logical and geometric paths when a user provides her/his preferences on spaces and geometric locations.

The following sub-subsections introduce routing options which indicate a user’s needs (e.g., visiting some SOIs and/or POIs in order) and the resulting path(s) for the user. As routing can be conducted in the logical and the geometric networks independently, this section presents the separate routing options for the two types of network. In addition, the semantics in the logical network and the geometry in the geometric network can be combined to increase the flexibility for routing. For example, routing results for the logical network can exclude unrelated spaces and then derive geometric paths more efficiently. Thus, seven routing options based on combinations of the two networks are introduced in this section.

To sum up, three categories of routing options can be identified according to a user’s needs: 1) only the logical network is used to compute logical paths for users; 2) only the geometric network is used to compute geometric paths for users; and 3) both the logical and geometric networks are adopted for routing (i.e., the two-level routing), and a user obtains logical and geometric paths. The next part elaborates these routing options and presents examples of them.

§ 3.4.1 Routing using the logical network

This section introduces routing options only with a logical network. In such cases, a user can find the way through spaces without geometric information. In other words, the user asks for only logical paths instead of geometric paths since the space sequences reflected by the logical paths are enough for her/him to follow. Commonly a user has no understanding about spaces’ semantics, but she/he can perceive the name of spaces. Thus, a user receives paths based on the names of the spaces to be visited. For example, a logical path is presented to a user as the following description: ‘I am in Office 260, I will pass by left corridor, follow down stair 1, and arrive at entrance hall’. In this description, the logical path consists of the spaces Office 260, left corridor, stair 1, and entrance hall. The logical path can be presented textually or graphically with highlighted spaces on a digital map or in a building model. In this thesis, the second approach is used.

To compute a logical path, a user needs to specify the start and target spaces. She/he can also specify some intermediate spaces (i.e., SOIs) to be visited between the start and target ones. The computed path would cross through the given SOIs in a given order. The following routing options are denoted with the prefix ‘L’ which means ‘Logical Path’.
L1.1 A user provides NO SOI. The user receives one logical path visualized as highlighted spaces on a digital map. The user specifies the start and the target space and requests a logical path. For example, the user is in the visitor reception (the start space) of a building and he wants to go to the conference centre (the target space). Then the user obtains a logical path referring to the names of the intermediate spaces from the visitor reception to the conference centre. Note that there may be multiple logical paths between two spaces. In such cases, the routing system would present the ‘best’ one (e.g., with the minimum number of spaces) for the user as the final path, which will be introduced in Chapter 4.

L1.2 A user provides SOIs to be visited and their ordering. The user gets one logical path through the given SOIs.

The user specifies a set of spaces and their order to be passed, and then asks for a logical path to cross through the given spaces. For example, the user is in a corridor of a faculty (the start space), and she/he wants to go to the lecture room Z (the target space). Before arriving at room Z, she/he would like to drop by the faculty’s library (i.e., a SOI). Then the user gets a logical path which crosses the library to room Z. The resulting logical path traverses the library and room Z sequentially.

Option L1.1 and option L1.2 are applied to different scenarios. Option L1.1 generates logical paths automatically for a user according to a specified criterion (e.g., the fewest spaces to be visited). The selection criteria of logical paths will be elaborated in Chapter 4. While option L1.2 applies for the users who have specific needs (e.g., visiting SOIs sequentially), and the users can add any number of SOIs in between the start and target spaces. In the example of option L1.2, the user can add more SOIs between the library and room Z. Actually L1.1 is the special case of L1.2 where the POI number is 0.

§ 3.4.2 Routing using the geometric network

Both the geometric and the logical networks can represent all the indoor spaces, some of the spaces, or only one space of a building. Based on a complete (all spaces) or partial geometric network (one or some spaces), geometric paths between two GeometricNodes are computed. The GeometricNodes lie in one space or different spaces. In one space, the path between two GeometricNodes is computed in the geometric network of the space. In the complete geometric network, a path can be computed between any two GeometricNodes. A geometric network can be created in selected spaces, and the geometric network can be built ‘on the fly’.

This section presents the routing options using just the geometric network. These options can be applied to either complete or partial geometric networks. In all the cases, a user needs to specify the start and target locations to get a geometric path. Also, she/he can set some POIs where the computed path needs to cross orderly. The following routing options are denoted with the prefix ‘G’ which means ‘Geometric Path’.

G2.1 A user provides NO POI and one size of the user. The user receives one geometric path, which is visualized on a digital map.

---

1 POIs not ordered are out of the discussion in this thesis.
Given the start and target locations, a user gets a complete geometric path. For example, the user is at the main entrance of a skyscraper which indicates the coordinates of the user. She/he wants to go to a clothing store in this building, and the target location is denoted by the coordinates of the store’s entrance. A geometric path (e.g., the shortest path) between the main entrance and the store’s entrance is computed.

**G2.2** A user provides POIs and their ordering\(^2\), and one size of the user. The user gets a geometric path through the given POIs sequentially.

This option offers users the possibility to specify POIs. A digital map is provided to a user to select POIs. As mentioned before, POIs refer to the coordinates related to the spaces which the user would visit (e.g., shops, coffee corners, toilets, benches, shelves, counters, etc.). The map contains POIs as the coordinates of the points close to or at these spaces. For example, a user at a hospital walks to the reception desk to make an appointment, then goes to a waiting section. After the user sees a doctor, the doctor tells the user to pass by the inquiry point on the same floor. Following the corridor, the user would make a right turn at the corridor’s end to the pharmacy counter to pick up medicine. In this example, the user specifies POIs related to the reception desk, the waiting section, the inquiry point, the corridor’s end and the pharmacy counter. The POIs are depicted by a point in the functional space close to the reception desk, a point inside the waiting section, a point close to the inquiry point, a point at the corridor’s end, and that at the front of the pharmacy counter. A geometric path is computed to cross through these POIs sequentially.

**G2.3** A user provides POIs with their order and different sizes of the user. The user gets a geometric path consisting of several parts. Each part is suitable for one of the given sizes.

This option enables the routing that provides paths according to a user size that is the diameter of the circumcircle covering the user and her/his operated objects. For example, in an airport a member of staff steps out of the office and walks to a cart-collection location, picks up a wheeled cart and transports goods with the cart to a port. The POIs are the office, cart-collection location and the port. From the office to the cart-collection location, the size is the staff’s dimension. While the user size changed when the staff picked the cart, i.e., the size is determined by the shape of the person with the cart, then the person needs to know the accurate path for the new size from the cart-collection location to the port. The geometric path consists of the path from the office to the cart-collection location, and the one from the cart-collection location to the port.

Option G2.1 is used to compute a geometric path between two locations for a given user. The two locations can be inside one space or in different spaces. In such a case, the user does not specify any POI and the resulting geometric path is suitable for the user size. Similar to the option L1.2 on a logical network, option G2.2 allows the user to specify a set of POIs to be visited and the order of these POIs. A geometric path is computed to pass through all the POIs in sequence according to the given user size.

The user size in the option G2.1 is considered constant. In contrast, the option G2.3 considers changes of a user size. Figure 3.18 provides the geometric paths for two dif-

\(^2\) POIs not ordered are not considered in this thesis. Also see footnote 1.
ferent user sizes in a floor plan. Figure 3.18a and b illustrate the geometric paths for the sizes 0.4 and 0.6 meter (m), respectively. The routing for geometric networks with different user sizes will be elaborated in Chapter 5.

In this research, a user needs to specify POI when changing sizes of the user are considered for routing. The POI is the location where the user size changes (e.g., the location where the user size increases because of picking a large tool).

Table 3.1 lists the routing options of the first and the second categories, which indicates whether each routing option includes POIs/SOIs and presents the number of user sizes assumed in each routing option. Network indicates the targeted network. Need SOI/POI? indicates whether a user needs to specify SOIs/POIs. Adopted user sizes refers to the number of user sizes.

Table 3.1 The comparison of user needs between the first and the second categories of routing options.

<table>
<thead>
<tr>
<th>Option</th>
<th>Network</th>
<th>Need ordered SOI/POI?</th>
<th>Assumed user sizes</th>
</tr>
</thead>
<tbody>
<tr>
<td>L1.1</td>
<td>Logical</td>
<td>No</td>
<td>None</td>
</tr>
<tr>
<td>L1.2</td>
<td>Logical</td>
<td>Yes</td>
<td>None</td>
</tr>
<tr>
<td>G2.1</td>
<td>Geometric</td>
<td>No</td>
<td>One</td>
</tr>
<tr>
<td>G2.2</td>
<td>Geometric</td>
<td>Yes</td>
<td>One</td>
</tr>
<tr>
<td>G2.3</td>
<td>Geometric</td>
<td>Yes</td>
<td>Multiple</td>
</tr>
</tbody>
</table>

§ 3.4.3 Routing using both networks

The previous two sections have explained the routing options using exclusively a logical or geometric network, while this section focuses on the combined use of the logical and geometric networks, which enables the routing computation to fulfil user needs on both of the networks. For example, a user specifies a SOI and a POI inside a building and she/he wants to get an accurate path. By routing for a logical network the user
obtains a logical path through the SOI, and then a partial geometric network is built ‘on
the fly’ based on the logical path. Finally, the user either obtains an accessible geometric
path which crosses through her/his POI, or receives a ‘No Path’ message.

It is necessary to further emphasize the distinction between SOIs and POIs for the rout-
ing options in this third category. A user can specify a space as SOI by its name (e.g.,
room 301). Thus, a SOI corresponds to a node of the logical network. As explained be-
fore, one space (e.g., SOI) can contain multiple POIs as points. SOIs are used for routing
on logical networks and POIs for routing on geometric networks.

Seven options are proposed for the two-level routing. For clarity, these options are
illustrated in a floor plan of Schiphol Airport, Netherlands. Logical networks are not
shown here. Logical paths are visualized by highlighting the related spaces that need
to be visited. Arrows indicate the visiting order for a user. For all cases, the user needs
to specify the start and target spaces and/or locations. Optionally the user can spec-
ify: 1) SOIs and/or POIs; and 2) the order that SOIs/POIs need to be visited; and 3) the
user size(s).

To clearly show the combinations of routing options of the first and the second catego-
rizes, the following routing options are denoted with the prefix ‘C’ which means ‘Com-

\[
\text{C3.1 A user provides NO SOIs and POIs, and a constant user size. The user receives one}
\]

logical path and one geometric path, which are visualized on a digital map.

This option makes use of options L1.1 and G2.1. Option L1.1 can result in several log-
ical paths depending on the path computation criterion. From these only one path is
selected (Figure 3.19b), a geometric network is built in the spaces indicated by the log-
ical path (Figure 3.19c). A geometric path (e.g., the shortest path) is computed for the
user (Figure 3.19d).

There is no accessible geometric path when one of the spaces is completely obstructed
by obstacles. In this case, the user would get a message about no paths and she/he
can decide to compute an alternative path. In the subsequent routing options, this ‘no
path’ case is treated in the same way.

\[
\text{C3.2 A user provides NO SOIs but specifies ORDERED POIs, and a constant user size.}
\]
The user receives one logical path and she/he gets a geometric path through the POIs
sequentially.

This option makes use of options L1.1 and G2.2. Option L1.1 provides one logical path
to the user to indicate POIs (e.g., two POIs in two spaces in Figure 3.20a). The geometric
path is computed through the POIs (Figure 3.20d).

Example: A student at the front door of a faculty building wants to visit a teacher’s office 080. According to the computed logical path the student needs to cross corridor L, staircase 3, passage T on another floor and then go to the office 080. The student knows there is an inquiry desk in corridor L, and she/he wants to go there and pick some brochure. Thus, the student adds the point in front of the inquiry desk as a POI, and then receives a detailed path through the POI and in the above spaces.
FIGURE 3.19 Illustration of option C3.1. (a) The start (in light purple) and target spaces (in light yellow); (b) A logical path (in green) is represented by a sequence of spaces, and arrows indicate the movement direction; (c) The geometric network in the spaces. Doors are nodes and black lines are edges; and (d) The geometric path.
FIGURE 3.20  Illustration of option C3.2. (a) The start and target spaces. The black points are two ordered POIs; (b) The logical path; (c) The geometric network in these spaces of the logical path; and (d) The geometric path through the POIs.
C3.3 A user provides ORDERED SOIs and POIs, and different user sizes (regarding specific SOI/POI). The users get one logical path through the SOIs and one geometric path for different sizes.

This option makes use of options L1.2 and G2.3. A user first gives the SOIs where the user’s size would change (Figure 3.21a). Then option L1.2 results in one logical path through the SOIs in order (Figure 3.21b). The user gets the names of the spaces to go through. Then in a digital map she/he specifies the POIs (the black point in Figure 3.21c) when the size changes. The geometric network is built in the spaces of the logical path. The resulting geometric path consists of several parts. These parts correspond to different sizes (e.g., the thin and thick lines in Figure 3.21d).

Example: As mentioned before, the staff in the airport wants to go to the cart-collection site and pick a cart (user size change), and then to the port to distribute goods. She/he would get one detailed path for both the initial size and the altered size.

C3.4 A user provides ORDERED SOIs and POIs, and a constant user size. The user gets one logical path through the SOIs and several separate geometric paths in these different SOIs.

This option makes use of Options L1.2 and G2.1. The user specifies the SOIs (e.g., the cyan space in Figure 3.22a) where she/he needs an accurate path. A logical path is computed through the SOIs (Figure 3.22b). Geometric networks are built separately in these SOIs (Figure 3.22c). The user provides POIs in each of the SOIs. Then option G2.1 is used to find a geometric path in each SOI (Figure 3.22d).

Example: A user in an airport knows the names of the spaces to go to. She/he also knows how to walk through most of the spaces. But in the large arrival hall (the SOI), the user needs details to confirm her/his direction. The user adds the opening between the lounge hall and the arrival hall as a POI, and specifies the arrival hall’s exit as another POI. A geometric path is computed in the SOI between the two POIs.
FIGURE 3.21 Illustration of option C3.3. (a) The start and target spaces, and the specified SOI (in cyan) and POI (where the user changes size); (b) A logical path indicated with arrows; (c) The resulting geometric network; and (d) The geometric path. The thin lines are for the initial size, and the thick lines for the altered size.
FIGURE 3.22 Illustration of option C3.4. (a) The start and target spaces. Except the presented SOI, the target space is also specified as a SOI; (b) The logical path through the SOIs; (c) Two separate geometric subnetworks in the SOI and the target space; and (d) Doors as POIs in the SOI and the target space, and related geometric paths.
C3.5 A user provides ORDERED SOIs but NO POIs, and a constant user size. The user gets one logical path through the SOIs and one geometric path. This option makes use of Options L1.2 and G2.1. A logical path is computed through the SOIs in order (Figure 3.23b). The geometric network is built in the spaces of the logical path (Figure 3.23c), and a geometric path (i.e., the shortest path) is computed (Figure 3.23d).

Example: A visitor at a skyscraper wants to visit several places (a clothing shop, a barbershop, a coffee house and a restaurant) in the given order. Similar to the cases above, a logical path through these places shows the user how to visit all the spaces sequentially (Figure 3.23b). The geometric path gives the user details for walking in these spaces (Figure 3.23d).

C3.6 A user provides ORDERED SOIs and POIs, and a constant user size. The user gets one logical path through the SOIs and one geometric path through the POIs. This option makes use of Options L1.2 and G2.2. A user is orientated with some important spaces (SOIs) (e.g., the cyan spaces in Figure 3.24a) and locations (POIs) in them (e.g., the black dots in Figure 3.24b). A logical path is computed through the SOIs (see Figure 3.24b). The geometric network is built in the spaces on the logical path (see Figure 3.24c). Then a geometric path is computed to pass through the POIs (see Figure 3.24d).

Example: In an airport, the user walks first to the departure hall and finds an inquiry desk. Then she/he needs to find a check-in desk in the departure hall. Afterwards, the user goes to another floor, and enters the inspection section to pass a security gate. Finally, the user arrives at the departure lounge. The departure hall, the inspection section and the departure lounge are the SOIs, while the inquiry desk, the check-in desk, and the security gate are the POIs. In such a case, the user not only gives the SOIs, but also specifies the POIs to go inside the SOIs.
FIGURE 3.23 Illustration of option C3.5. (a) The start and target spaces, and the SOI; (b) The logical path through the SOI; (c) The geometric network in the spaces of the logical path; and (d) The geometric path.
FIGURE 3.24 Illustration of option C3.6. (a) The start and target spaces. The two cyan spaces are the ordered SOIs; (b) The logical path through these SOIs orderly; (c) The geometric network in the spaces of the logical path; and (d) The geometric path through the three POIs in order.
C3.7 A user provides ORDERED SOIs but NO POIs, and a constant user size. Based on a computed geometric path and its related logical path, the user does not approve or is not allowed to pass through one or more spaces (named Anti-SOI). Then she/he specifies SOIs and gets one logical path and one geometric path.

This option makes use of Options G2.1 and L1.2. The user first obtains a geometric path between the start and target locations (see Figure 3.25a) from the routing on the complete geometric network of a building. If the user does not satisfy the geometric path due to some spaces crossed by the geometric path, she/he can specify SOIs (see Figure 3.25c) and get a corresponding logical path (see Figure 3.25d). Then a geometric network is built in the spaces on the logical path (Figure 3.25e). The related geometric path is computed for the user (see Figure 3.25f).

Example: In a skyscraper the user gets a geometric path from office 1010 to the entrance hall on the ground floor, but the user is not satisfied with the space stair K crossed by the geometric path. With a smart phone visualizing the logical network of the skyscraper, the user manually adds the space elevator L as a SOI and gets the logical path through the elevator L. Then the new geometric path is computed for the user.
(a)

(b)

(c)

(d)
FIGURE 3.25 Illustration of option C3.7. (a) A computed geometric path for a user; (b) The corresponding logical path of the geometric path; (c) Anti-SOI (in purple) of the user, and the SOI (in cyan) for new computation; (d) A new logical path through the SOI; (e) The geometric network in the spaces of the new logical path; and (f) The final geometric path.
In the options from C3.1 to C3.6, a routing option is applied first on the logical network and then another routing option on the geometric network. In this way, only a partial geometric network of a building is needed for routing. The partial network is bound to the computed logical path (e.g., built in the spaces on the logical path) and a user’s decision (e.g., option C3.4, built only in the user specified spaces). The partial network may not result in a path, which can be supplemented by re-routing. In contrast, option C3.7 applies first option G2.1 on a complete geometric network of a building, then computes a user-preferred logical path on the logical network. Option C3.7 is a special case that allows a user to adjust geometric paths on demand.

Without SOIs, option C3.1 can provide a geometric path through all the spaces on a logical path. A user can also decide to compute geometric paths in only a part of the spaces (i.e., option C3.4). Option C3.4 and C3.5 both use options L1.2 and G2.1. In the two options, a user gets a logical path through the selected SOIs. In option C3.4 the user obtains geometric paths only in the SOIs, but option C3.5 provides a geometric path through all the spaces on the logical path. This is the main difference between option C3.4 and option C3.5.

Table 3.2 compares user needs of all the above options, including specifying POI or SOI (Need ordered SOI? and Need ordered POI?) and the number of user sizes (Adopted user sizes).

<table>
<thead>
<tr>
<th>Option</th>
<th>Combined options</th>
<th>Need ordered SOI?</th>
<th>Need ordered POI?</th>
<th>Adopted user sizes</th>
<th>Resulting geometric path</th>
</tr>
</thead>
<tbody>
<tr>
<td>C3.1</td>
<td>L1.1 + G2.1</td>
<td>No</td>
<td>No</td>
<td>One</td>
<td>Yes</td>
</tr>
<tr>
<td>C3.2</td>
<td>L1.1 + G2.2</td>
<td>No</td>
<td>Yes</td>
<td>One</td>
<td>Yes</td>
</tr>
<tr>
<td>C3.3</td>
<td>L1.2 + G2.3</td>
<td>Yes</td>
<td>Yes</td>
<td>Multiple</td>
<td>Yes</td>
</tr>
<tr>
<td>C3.4</td>
<td>L1.2 + G2.1</td>
<td>Yes</td>
<td>Yes</td>
<td>One</td>
<td>No</td>
</tr>
<tr>
<td>C3.5</td>
<td>L1.2 + G2.1</td>
<td>Yes</td>
<td>No</td>
<td>One</td>
<td>Yes</td>
</tr>
<tr>
<td>C3.6</td>
<td>L1.2 + G2.2</td>
<td>Yes</td>
<td>Yes</td>
<td>One</td>
<td>Yes</td>
</tr>
<tr>
<td>C3.7</td>
<td>G2.1 + L1.2</td>
<td>Yes</td>
<td>No</td>
<td>One</td>
<td>Yes</td>
</tr>
</tbody>
</table>

Table 3.2 compares user needs of all the above options, including specifying POI or SOI and the number of user sizes. Option C3.1 does not need SOIs and POIs, while the other options take user intervention on SOIs and/or POIs. Considering the influence of a user size on routing, option C3.3 provides paths in terms of changes of the user size during a motion (the value ‘multiple’). The other options all consider one user size only.

These routing options are applied to different situations where indoor paths are not unique. A user’s preferences (the size and preferred SOIs/POIs) result in the suitable path(s). Table 3.2 is not an exclusive list of the routing options using both logical and geometric networks. Given other application scenarios, more options can be proposed based on the combination of routings on both logical and geometric networks. For example, a new option can be devised by changing the option C3.2, where multiple user sizes could be adopted.
§ 3.5 Summary

This chapter responded to and explained the following research sub-questions in Chapter 1:

2. What data and navigation model is appropriate to represent the semantics, topology and geometry of indoor spaces?

3. What kind of user-related paths can be computed with the semantics, topology and geometry of indoor spaces?

For question 1, though IndoorGML provides a description of specific semantics, topology and geometry of indoor spaces, it does not aim to generate navigation networks. Therefore, in this chapter INSM is proposed. The INSM semantics can be projected onto both simple and complex buildings with different subdivisions, which can facilitate the generation of navigation networks. Semantics in INSM, such as NavigableUnit (NU), VerticalUnit (VU), HorizontalUnit (HU), Opening (OPN) and Obstacle (OBS), are proposed according to the navigational functionality of indoor spaces. Basic measures of space such as Name, Bottom Height, and Top Height are designed for related classes in INSM. Connectivity of spaces (topology) is explicitly stored in INSM, which can derive logical networks automatically. Building geometry is stored in INSM and it can be directly used for the creation of geometric networks. In this thesis, the geometry of indoor space is represented by 2D/3D surfaces, such as 2D surfaces for HU and 3D surfaces for VU.

Regarding question 2, the two-level routing approach is proposed which adopts two types of navigation model, i.e., the logical and geometric networks. The user-related paths are defined by a user’s motion ability, size and space/location preferences etc. Based on the two types of network, the two-level routing approach aims to provide customized paths for a user when the user specifies her/his preferences on indoor spaces and/or locations. INSM is used to facilitate the derivation of the logical and the geometric networks (see Section 3.3). The logical network should be suitable for a user’s motion ability (e.g., wheelchair users). The two-level routing approach integrates routing on both the logical and the geometric network and seven routing options are designed for different applications (see Section 3.4). These routing options allow indoor routing to be flexibly computed according to user needs such as passing through ordered SOIs/POIs and obstacle-avoidance. The resulting geometric paths are always accessible to users with the given size. Note that the seven options are not an exhaustive list of the two-level routing, and it is possible to devise more options for other user-related applications. For example, a user specifies SOIs and POIs which are not ordered. In this case, the two-level routing may provide a group of geometric paths for the user.

This chapter has also introduced the other two categories of routing options (see Section 3.4). Two routing options are based on using just the logical network and three options based on using just the geometric network, respectively. This chapter has not mentioned yet the implantation of all these routing options, which will be introduced in chapters 6 and 7.
Regarding the possible ‘NO Path’ case in the two-level routing, the user would be informed by a message about no paths, and the user can request an alternative path. First, a new logical path is computed for the user, and then the corresponding geometric path is also provided on demand. In addition, a user may get multiple logical/geometric paths from the two-level routing. In this case, the routing system would pick one of the ‘optimal’ paths for the user, which can reduce the user affordance for using the system.

The subsequent chapters will further explain details of the proposed routing options. Chapter 4 will introduce routing criteria regarding space semantics and human wayfinding behaviours, and elaborate the routing computation on the logical network (i.e., on the abstract level). Chapter 5 will present the routing computation on the geometric network (i.e., on the detailed level), which makes it possible to compute accessible geometric paths according to a given user size. Chapter 6 will present a realization of one-level routing, i.e., routing on just the logical or the geometric network. Chapter 7 will present realization of the two-level routing.
4 Routing on logical networks

The previous chapter introduced the two-level routing approach which follows a Coarse-to-Fine routing manner. Logical paths are computed on the logical network (i.e., on the abstract level), which provides a general picture about motions passing through indoor spaces. Such paths are conceptual to pedestrians without geometric details. In some cases, given a start and a target space, there are multiple logical paths for a user. For example, a visitor at the main hall of a building can take either of two elevators/stairs to arrive at the same office on the top floor, which corresponds to different logical paths. Therefore, it is necessary to compute qualified logical paths according to user demands on space.

This chapter presents criteria for routing on the logical network, and introduces the routing procedure. Section 4.1 shows the purposes of routing on the logical network. Section 4.2 presents the process of logical network derivation from the INSM. Inspired by human wayfinding behaviours, three existing strategies are adopted and six routing criteria are proposed to simulate the wayfinding results of pedestrians (Section 4.3). The criteria involve different constraints on an indoor path, such as minimizing the NavigableUnit (NU), HorizontalConnector (HC) or VerticalUnit (VU). These criteria are defined on the INSM semantics and their meanings are given in Section 4.3. Section 4.4 presents the concrete steps of routing with a single criterion, and with multiple ordered criteria. With a single criterion, the logical network is weighted according to the semantics of logical nodes and then routing is conducted by using the Dijkstra algorithm [Dij59]. In addition, routing with the ordered multiple criteria can reduce the number of logical paths between two spaces. This chapter, which closes with a short summary (Section 4.5), is related to the following author’s own publications: [LZ13a].

§ 4.1 Motivation

Two distinct users (e.g., pedestrians and wheelchair users) obtain different subdivision results for the same building [BNK09]. As a logical network is derived from the building’s subdivision result, the two users will navigate in different logical networks. For a given building, the user’s locomotion type [KK12] and other constraints on spaces (e.g., access permission) can be used to determine spaces which a user can locate and navigate to [BNZK13]. Accordingly, the two users would get different paths under the same routing criterion (e.g., the minimum number of spaces to be passed). In addition, a user may have a specific preference on indoor paths with respect to a given scenario. For example, a path from the entrance hall to an office on the fifth floor in a building can be with or without the use of the elevator.

The semantics of spaces is useful for the description of paths. Pedestrians cannot precisely perceive a detailed geometric path by metric instructions. Precise distances (e.g., 40 or 45 meters) can be measurable for robots, but the subtle difference between the distances cannot be distinguished very well by pedestrians. Thus, metric instructions, such as ‘Turn sharp right, walk for 9 metres and reach the north stairs’ [RZC14], should be replaced by other semantic descriptions. For instance, Rehrl (2007) has applied semantic instructions to provide references to the semantics of indoor environments,
such as ‘Walk to the end of the corridor, meet at the bottom of the stairs, and walk up the stairs’ [RGL+07]. Here the corridor and the stairs are semantics of indoor spaces.

A pedestrian’s preference for specific spaces can be reflected with semantics. For example, one needs a path to pass through the minimum number of corridors. In this case, the user preference for space is described by the number and the semantics of spaces. Similarly, a pedestrian prefers to pass through the minimum number of ordinary spaces to a destination, when a large building is subdivided into spaces with a similar size. This user has no specific need for space semantics but she/he expects the resulting path would be a short route.

In this research a user is asked to provide a similar preference description. All preferences collected from the user are sorted in a priority order. For instance, two ordered preferences are ‘the minimum number of passed spaces’ and ‘to minimize the use of elevators’. Following the first preference, ‘the minimum number of spaces’, it could happen that the path crosses a stair or that there are several possible paths. Then the second preference is considered and the user is provided with a path via an elevator, which however can consist of more spaces than other path choices (without an elevator). Therefore, routing with pedestrian preferences does not ensure the shortest path.

Logical networks are often used to give an abstract representation of a building [ICC12] and support the generation of human-readable descriptions about moving through the building. According to the wayfinding theory, there are three types of knowledge supporting a person’s cognitive maps, i.e., survey, path, and landmark knowledge [SW75, TG83]. Survey knowledge is the understanding of the topological structure of an environment. Path knowledge is about the method of finding the way from a start to a destination via many intermediate locations. Landmark knowledge means finding the way with locations with high salience. ‘Landmark’ means the distinctive objects in a navigation environment such as high-rise buildings. Compared to survey knowledge (e.g., overview of indoor maps) about a building, path knowledge (e.g., steps to find a destination) better serves a user for indoor wayfinding [THR82]. Commonly, humans describe paths to each other using a logical path. For example: ‘from the entrance hall on the ground floor, go up the stair to the second floor, turn left, and go to the end of the corridor’. In this example, the logical path is represented by the sequence of the entrance hall, the stair and the corridor. Also, landmark knowledge helps the user to orient and better follow the described path. Landmarks can help pedestrians to be aware of the route being followed [FLZS12]. Salient spaces acting as landmarks in a walking environment can also support users’ orientation. For example, the entrance hall of a building is a salient space. A user can always step back to the hall whenever she/he gets lost in the building, and restart the walking from there.

As mentioned in Chapter 2, wayfinding is a kind of self-guidance to target locations with the help of tools (e.g., signage) [SKO97], which can be considered a heuristic process. Hölscher et al. [HB07] mention three strategies of pedestrian wayfinding behaviour that aims to simplify the heuristic as much as possible: 1) pedestrians approach the destination by following the same floor as much as possible and then taking the closest stairs to the destination; 2) pedestrians arrive first at the floor of the destination as directly as possible, and then go horizontally to the destination; and 3) pedestrians always pass high-salience indoor areas. It is argued that the three strategies can reduce the memory load on users during wayfinding. If one computes such a path for users, i.e., the users are aware of the spaces to be passed and the sequence of
the spaces, they can find the desired space by following only signage or verbal instructions (if available).

To simulate such wayfinding behaviours, this research defines appropriate routing criteria. For example, one computes a logical path related to the minimum number of visited spaces, because a user is inclined to distort her/his orientation after switching too many spaces. Figure 4.1 presents an example of two floors with different subdivision results (in a front view). The top floor (in light green) has fewer spaces than the ground floor (in dark green). Walking on the top floor would be easier for a user since she/he needs to memorize fewer spaces. Thus, logical paths derived by these criteria support the users who need a general path without the burden of details, and they can follow the general path with the help of signage or visualization on a mobile device.

![Figure 4.1](image)

**FIGURE 4.1** Two floors of a building contain a different number of spaces. The black arrows indicate the moving directions of two stairs (VU).

Each criteria is reflected by a weighted logical network, and a logical path is derived by minimizing the weights attached to each node of the logical network. To compute logical paths with multiple defined criteria, an optimization approach named Lexicographical Goal Programming [J10] is adopted for routing (see Chapter 2). Section 4.2 presents the derivation process of the logical network. Section 4.3 introduces the routing criteria utilizing INSM semantics presented in Chapter 3, and Section 4.4 presents the computation of logical paths.

### § 4.2 Logical network derivation

Before the routing criteria are presented, this section introduces the derivation of a logical network from INSM. As mentioned in Chapter 3, Classes of *NavigableUnit* (NU) and *Opening* (OPN) are associated in *INSM*, which reveals the connection between NU and OPN. For instance, a door (OPN) links two other spaces (NU). Accordingly, the connection between the two spaces can be automatically confirmed (i.e., edge of the logical network). In this way, all the connections can be confirmed regarding all the NUs in the
FIGURE 4.2 Derivation of a logical network. (a) Connection between OPN and NU; (b) Edges of a logical network.

building. The logical network consists of all the NUs (nodes) and these connections (edges). The nodes inherit the semantics of the spaces from INSM.

Figure 4.2 shows a simple example of a logical network derivation. NU represents the nodes of the logical network, and the connection between NU and OPN determined from INSM (D1 with S1 and S2, and D2 with S2 and S3 in Figure 4.2a). Then the edges of the logical network are inferred (Figure 4.2b).

Normally NUs derived from a specific subdivision would not change. As nodes of the logical network represent NU of a building, thus the logical network can generally be considered fixed. But the edges of the logical network can be influenced by an access permission issue, e.g., some NUs close due to a time restriction. In such a case, the logical network needs to be updated. In brief, for stable buildings a logical network can be created once and then stored for routing, but the logical network can also be created ‘on the fly’ according to the restriction on spaces.

§ 4.3 Routing criteria based on INSM semantics

Logical paths can be readily computed considering the semantics of indoor spaces. Taking INSM semantics as an example, a logical path needs to consider VU as the first priority when a user has to go to a different floor in a building. With more detailed semantics of indoor spaces such as a kiosk, a logical path can be computed using the functional use of the spaces. For instance, a user can ask for a logical path, which passes close to many kiosks.

This research adopts INSM semantics for routing on logical networks, which is based on three reasons: 1) The INSM semantics provides general navigational functions (i.e., horizontal or vertical, Connector or End) of indoor spaces; 2) the INSM semantics can be easily derived for different buildings in 2D and 3D data (e.g., CAD floor plans, CityGML LoD4 and BIM IFC); 3) by using INSM, semantics can be assigned to any subdivision result of a building. In addition, the INSM can ideally depict complex indoor environments such as intermediate levels (a platform inside a large hall, see Figure 4.3). In
Figure 4.3, the semantics of each space indicate that the END (platform) is contained in the HC, and the VC-VU-VC connection links the END and the HC vertically.

![Figure 4.3](image)

**FIGURE 4.3** An example of a complex building depicted by INSM. (a) The platform in the middle of the large hall; and (b) the INSM semantics of spaces in this scenario.

To reduce the ambiguity in navigation on a logical network, six routing criteria are proposed (see below). The logical network is weighed according to each proposed criterion and minimizes the edge weights of the logical network according to the routing criterion. The proposed routing criteria aim to minimize node/edge weights to form a logical path. The weight of a node can indicate the importance of the node in the logical network for a specific purpose. Six criteria on logical network are designed from INSM semantics. Paths can then be computed with regard to each single criterion or by applying combinations of them.

These criteria are presented as follows:

- **Minimum NavigableUnit (NU).** The criterion derives a path with the minimum number of traversed spaces (i.e., NU). This means a user goes to a place by passing through as few NUs as possible.

- **Minimum HorizontalConnector (HC).** This results in a path with the minimum number of HCs.

- **Minimum VerticalUnit (VU).** The Minimum VU derives a path minimizing the number of VUs in a logical path.

- **Central HorizontalConnector (HC).** Except for the start and target nodes, the Central HC results in a logical path preserving the high level of accumulated centrality of HCs and is also as direct as possible to the target. In a network the centrality of a node represents the accessibility and importance of the node to the other nodes [S.05].

- **HorizontalConnector (HC) Prior.** The HC Prior criterion results in a logical path prioritizing HC nodes and which is also as direct as possible to the target.

- **VerticalUnit (VU) Prior.** This criterion generates a logical path prioritizing VU nodes and which is also as direct as possible to the target.
The first three criteria (Minimum NU, Minimum HC and Minimum VU) are about the minimization of NU, HC and VU nodes in a logical path, respectively. The Central HC criterion provides a direct path (no detours) to the destination with high accumulated centrality of HC’s. The Central HC path will have the higher accumulated centrality of HC’s if there are other logical paths including the same number of spaces. The HC Prior and VU Prior paths are applied when HC/VU nodes have precedence to form a direct path to the destination.

The central HC evaluates the centrality of HC nodes in the logical network. There are different definitions of centrality such as degree, betweenness and closeness. The degree of nodes indicates the number of nodes connecting with a node. The closeness is the average length of the shortest paths between the node and all other nodes in the same graph [G.66, BV13]. In this sense, a node is ‘central’ when it is close to all the other nodes. The betweenness of a node is the ratio of the number of shortest paths via the node and that of all the shortest paths (among all the possible pairs of start and target nodes) [Bra01]. This thesis adopts the degree as the value of centrality for a logical network since it refers to the choices for the next step.

The above concepts are illustrated with the following example. Figure 4.4 presents an artificial building in the front view. Figure 4.4a presents a minimum NU path including the fewest number of spaces passed by a user. Figure 4.4b shows the logical network with a minimum VU path through only two VUs between the start and the destination. Figure 4.4b presents the stairs only; the VU class has three subtypes: Escalator (ES), Stair (ST) and Elevator (EL). Thus the minimum VU and VU Prior criteria have more derivatives, i.e., minimum ES, minimum ST, minimum EL, ES Prior, ST Prior, and EL Prior. The computation about these paths will be further explained in Section 4.4.
4 Routing on logical networks
FIGURE 4.4  Illustration of four criteria in an artificial building. (a) A minimum NU path; (b) A minimum VU path; (c) A minimum HC path; and (d) An HC prior path. The HC prior path includes more HCs (HC9, HC10, HC11, HC12, HC13 and HC14) than the minimum HC path.
The minimum HC path ensures that a user can pass as few HCs as possible when she/he transits between the floors of the start and target spaces. Figure 4.4c presents a minimum HC path where the user transits stairs and passes only four HCs (HC4, HC5, HC20 and HC21) to arrive at the target.

Figure 4.4d presents an example of a HC prior path that crosses eight HCs (HC4, HC5, HC14, HC13, HC12, HC11, HC10 and HC9) to reach the target. Compared to the minimum HC path, the HC prior path includes more HCs and fewer VUs. In contrast, the minimum HC path has the fewest HCs, irrespective of the number of VUs.

FIGURE 4.5 Illustration of the strategies of the floor, flat location and SOI.

The pedestrian wayfinding strategies mentioned by Hölscher et al. [HB07] can be realized by using the criterion minimum NU. In this thesis the three strategies are named as floor, flat location and SOI strategies. The floor strategy finds first the vertical position (i.e., the floor) of the target space irrespective of the horizontal position of the target, while the flat location strategy aims to reach the approximate horizontal location of the space as directly as possible and regardless of switching floors. In this thesis the horizontal location is regarded as the closest VU space(s) to the target space on the target floor. The SOI strategy covers specified space(s) to be visited. These salient spaces act as landmarks [FLZS12] to the user. When the start and target spaces are on the same floor, the logical path does not need to be computed with the floor and flat location strategies, because both the strategies are related to paths crossing floors. Figure 4.5 provides examples of the paths conforming to the three strategies.

Each of the three strategies are denoted as several segmented minimum NU paths. The common ground among them is that the three strategies need segmented paths through one or more intermediate nodes in a logical network. Their main difference is in the choices of the intermediate nodes. The strategies of floor and flat location automatically specify the intermediate nodes by algorithms, while users need to specify intermediate nodes in the SOI strategy.
To implement the floor strategy, a simplified network of the original logical network is used to select the key VU. The simplified network consists of the start and target nodes and all the VU nodes (see Figure 4.6b). Each edge of the simplified logical network represents a number of HC and/or VC between the two nodes, and the weight of the edge is the number of these in-between the HC/VC nodes. The computation for a floor strategy path is presented in the following algorithm, which is illustrated in an artificial building (Figure 4.6):

**Algorithm 1** Implement floor strategy in the logical network for routing.

**Input:** Logical network $Net_l$, the start node $n_s$, the target node $n_t$, the node set $VU_{node}$ of all VU, and the simplified logical network $Net_s$ consisting of just $n_s$, $n_t$ and $VU_{node}$. Each edge of $Net_s$ is weighted with the number of the in-between NU of the two nodes.

**Output:** The floor strategy path $p$.

1: procedure FLOORSTRATEGYPATH($Net_l$, $Net_s$, $n_s$, $n_t$, $VU_{node}$)
2: from the set $VU_{node}$, select nodes $VU_f$ on the same floor to the target.
3: for each node $vu_i$ in $VU_f$ do
4: compute the shortest path on $Net_s$ from $n_s$ to $vu_i$
5: record the total weight of the above shortest path to a value set $W$.
6: end for
7: locate the node $vu$ with the lowest weight in $W$
8: compute the Minimum NU path $p_1$ from $n_s$ to $vu$ on $Net_l$
9: compute the Minimum NU path $p_2$ from $vu$ to $n_t$ on $Net_l$
10: aggregate $p_1$ and $p_2$ into the floor strategy path $p$
11: return $p$
12: end procedure
FIGURE 4.6 Illustration of the computation of the floor strategy path. (a) A building in the front view. The start and target nodes are HCS and HC19; (b) The simplified logical network. VU6 on the target’s floor; (c) The floor strategy path in the simplified network; and (d) The floor strategy path in the original logical network.
For the *flat location* strategy, the simplified network of the original logical network is also used. The computation for *flat location* strategy path is presented in the following algorithm, which is also illustrated in the above artificial building (Figure 4.7):

Algorithm 2 Implement *flat location* strategy in the logical network for routing.

**Input:** Logical network $Net_l$, the start node $n_s$, the target node $n_t$, the node set $V_{node}$ of all $Vu$ and the simplified logical network $Net_s$ consisting of just $n_s, n_t$ and $V_{node}$. Each edge of $Net_s$ is weighted with the number of the in-between $NU$ of the two nodes.

**Output:** The *flat location* strategy path $p$.

1: procedure FlatLocationStrategyPath($Net_l, Net_s, n_s, n_t, V_{node}$)
2: for each node $vu_i$ in $V_{node}$ do
3: compute the shortest path on $Net_s$ from $vu_i$ to $n_t$
4: record the total weight of the shortest path to a value set $W$.
5: end for
6: locate the node $vu$ with the lowest weight in $W$
7: compute the Minimum NU path $p_1$ from $n_s$ to $vu$ on $Net_l$
8: compute the Minimum NU path $p_2$ from $vu$ to $n_t$ on $Net_l$
9: aggregate $p_1$ and $p_2$ into the *flat location* strategy path $p$
10: return $p$
11: end procedure
FIGURE 4.7 Illustration of the computation of the flat location strategy path. (a) The building in the front view. The start and target nodes are HC5 and HC19; (b) The simplified logical network. VU9 is the closest node to the target; (c) The flat location strategy path in the simplified network; and (d) The flat location strategy path in the original logical network.
The SOI strategy allows the user to select nodes in the logical network in an order. In such a case, a *minimum NU* path is computed between every two specified nodes sequentially (see Figure 4.5). All these *minimum NU* paths form the complete SOI strategy path.

Comparing the *floor* and *flat location* strategies, one can find that the two strategies first select the VU to the target and then compute related paths via the VU. However, the *floor strategy* selects the VU on the same floor of the target and closest to the start; the *flat location strategy* selects the VU closest to the target.

Additionally, a user can specify the subtype (*i.e.*, *EL, ST*, or *ES*) of VU in the *floor* and *flat location* strategies. For example, if a user specifies the subtype as *EL*, then a *floor/flat location* path is computed for the user by selecting only *ELS*. 

### § 4.4 Routing procedure

Section 3.4.1 has introduced two routing options for the logical network. Option L1.1 is that a user provides NO SOI, while option L1.2 is the user specifies both SOIs and their ordering. The six proposed criteria (i.e., *Minimum NU, Minimum HC, Minimum VU, Central HC, HC Prior*, and *VU Prior*), and the presented *floor strategy* and *flat location strategy* (i.e., multiple segmented *Minimum NU* paths) are applied to option L1.1. In these cases, a user can receive a logical path or ‘No Path’ message from path computation by selected criteria/strategies. The *SOI strategy* (i.e., multiple segmented *Minimum NU* paths via user-specified nodes) is applied to option L1.2, because the group of ordered nodes specified by users in this strategy are the specified SOIs. The rest of this section will present path computation with these criteria. Then paths from the three presented strategies [CTG+06] can be computed based on the *Minimum NU* criterion.

As the logical network is basically a graph, therefore these criteria have to be presented as weights to the edges or the nodes. Subsection 4.4.1 firstly introduces how to define edge weights of the logical network for different criteria. Then Subsection 4.4.2 presents routing computation with a singular criterion, and elaborates the path computation when multiple criteria are needed.

### § 4.4.1 Weighted routing

The nodes of the logical network hold the semantics of the spaces which define the priority for navigation. In a logical network, edges indicate connectivity among nodes but nothing more. A logical path represents a sequence of spaces to be passed. Thus weights are assigned to logical nodes first, then transmit these weights to logical edges. Consequently, the weight of an edge reflects the significance of its emitting node (see Figure 4.8).

Routing with the criterion is to minimize the edge weights. In this way, graph-based algorithms are employed to obtain logical paths. For the purpose of minimization, the following values are used: 0 means the node can be either in or not in a path, which is neutral (does not affect the result) to the path computation; a relatively large value such as 1000 means a user tries not to adopt the node in a path; and a lower value such
as 1 means it is a normal node and the number of nodes in the given type (e.g., HC) is preferred and would be counted in a path.

As shown previously, routing on the logical network relies only on HC, VC and VU nodes with the start and the target nodes. END nodes are not considered when routing is performed on the logical network. Therefore no weights are set for them. For each criterion, node weights are chosen in the light of node semantics. Table 4.1 presents all the node weights for the six proposed criteria and their variants (e.g., the Minimum EL is a variant of the Minimum VU). All the criteria are separated into two groups: 1) minimizing specific type; and 2) setting priority to specific types (see Table 3.2). The first group aims to minimize the given space type(s) (e.g., HC) in a logical path, regardless of the other types of nodes (e.g., VC & VU); the second group sets the priority for the given space type(s) and also averts the other space types as much as possible for a path.

The first group (i.e., minimizing specific type) includes minimum NU, minimum HC, minimum VU, minimum EL, minimum ES, minimum ST, minimum EL&ES, minimum EL&ST, and minimum ES&ST (see Table 4.1).

The criterion of minimum NU indicates that all nodes in a logical network are equally important, because each step for a user is to pass through a space. Thus 1 is set for all the nodes. The choices of VUs are not important for the criterion of minimum HC. Thus 0 is set for all the VC and VU nodes, while 1 is set for the HC nodes. For the same reason 0 is set to all the VC and HC values in the logical network for the minimum VU path, and 1 for all the VU nodes (EL, ES, and ST). The minimum VU criterion has six variants including minimum EL, minimum ES, minimum ST, minimum EL&ES, minimum EL&ST and minimum ES&ST. In these variants one or two subtypes of VU are emphasized, which means the subtypes are the first choice as VU for a path. Similar to the minimum VU, the six criteria set all the VC and HC values to 0; and the emphasized subtypes of VU are set to 1, while the other subtypes of VU are set to 1000 (see Table 4.1). For example, in the minimum EL criterion, the emphasized subtype is EL and then all the EL nodes are set to 1. The rest of the subtypes, i.e., ES and ST, are all set to 1000. In contrast, for the minimum ES&ST criterion, all the EL nodes are set to 1000 and the ES and ST nodes are set to 1, because ES and ST are the emphasized subtypes (see Table 4.1).
The second group (i.e., setting priority to specific types) includes: central HC, HC Prior, VU Prior, EL Prior, ES Prior, ST Prior, EL&ES Prior, EL&ST Prior, and ES&ST Prior (see Table 4.1). For central HC, 1000 is assigned to VC and all the VU nodes, which avoids adopting many VC/VU nodes. The HC nodes are set to the value subtracting their centrality from 1000. Because the central HC criterion aims to collect the high accumulated centrality of nodes in a direct path to the target by minimizing weights (see Section 4.3), the weight (centrality value) is adapted to the monotonicity of the minimization (i.e., to the lower accumulation) by subtracting the centrality of the HC nodes from the large number 1000.

HC Prior relates to horizontal movements. Thus 1 is set to the HC nodes (with high priority in a path) and 1000 to the other nodes (with the lower priority). Except for the central HC and HC Prior, the others in the second group concentrate on vertical movements. Similarly, 1 is set to nodes of the specified subtype(s) of VU, and 1000 to HC, VC nodes and nodes with the other subtypes of VU (see Table 4.1).

**Algorithm 3** Derive edge weights from node weights in a logical network

**Input:** A logical network G, the values Ws of node weights.

**Output:** The logical network G where the edges have been weighted in light of Ws.

1. **procedure** DERIVEWEIGHTS(G, Ws)
2. **for** each node in G **do**
3. find the edge set Es emitted from the node
4. look up the weight val of the node in Ws
5. **for** each edge e ∈ Es **do**
6. the weight of e is set as val
7. **end for**
8. **end for**
9. **return** G
10. **end procedure**

**FIGURE 4.8** Derivation of edge weights from those of nodes in a logical network. (a) Three rooms; (b) Weights of nodes; and (c) Transition to edge weights.

As mentioned above, the node weights are transformed to edge weights and then the Dijkstra algorithm [Dij59] is applied. Given a node property, Algorithm 3 presents the method that converts the node values to those of related edges. For each node the algorithm locates first the outcoming edges of the node, and picks the node’s value. Subsequently, it sets the weights of all the outcoming edges with the node’s value. Figure 4.8a presents three rooms, and Figure 4.8b presents their logical network and the
attribute values \(i.e.\ m, n\) and \(t\) of the nodes. Figure 4.8c presents the derived edge weights in the logical network.

### § 4.4.2 Multi-criteria routing

As discussed previously, a logical network is tailored for each specific user. Firstly, accessibility to all the edges of the logical network is confirmed. Secondly, the time restriction on all the edges will be investigated as well. Next, the network is simplified by removing all the nodes which are not accessible. These inaccessible nodes and related edges are removed in a given period. Then the remaining logical network is what is used for path computation. Figure 4.9 provides an example of time constraint on accessibility in a floor plan. This is the ground floor of the Architecture Faculty building at the Delft campus. The spaces of one section are inaccessible to visitors after 6 p.m. every day. Thus the related nodes and edges are removed (see Figure 4.9). Generally users with different authorizations obtain distinct logical networks.

![Figure 4.9](image)

**FIGURE 4.9** Time constraint on accessibility to a logical network. The logical network is embedded in spaces. (a) The original logical network; and (b) The modified logical network where the nodes and edges of a section are removed after work time.

Multiple logical paths may be derived with just one single criterion. All the resulting paths can be reported if they are equal according to a singular criterion. For example, in a building a user gets two minimum NU paths from the entrance hall to an offices via
two separate stairs. The user can select one of them based on the visualization of both paths.

To find out a subset of the ‘better’ paths for users among a number of alternatives, multiple criteria are employed in sequence (the 1st, 2nd, 3rd...) in routing according to the Lexicographical Goal Programming [JT10]. The users can specify multiple criteria in different priority orderings. Thus a priority ordering of criteria is leveraged to compute logical paths.

Algorithm 4 Compute logical paths with ordered criteria in a priority list on a logical network

**Input:** A prioritized list of ordered routing criteria \( F \), a logical network \( N \).

**Output:** The set of logical path(s) \( Sp \).

1. procedure **MULTICRITERIAROUTING**\( (F, N, Sp) \)
2. for criterion \( f_i \) in order in \( F \) do
3. if \( Sp \) is empty then
4. compute paths in \( N \) according to \( f_i \)
5. add the paths into \( Sp \)
6. else
7. compute paths in the paths of \( Sp \) according to \( f_i \)
8. replace the \( Sp \) by the new paths
9. end if
10.
11. if \( Sp \) contains only one path then
12. return \( Sp \)
13. end if
14. end for
15. return \( Sp \)
16. end procedure

![FIGURE 4.10](image)

The workflow of routing with priority ordering of criteria.

Algorithm 4 presents the computation process with a priority order of multiple criteria. By following the Lexicographical Goal Programming [JT10], this procedure is to compute a set \( Sp \) including all the semantic paths with the first criterion (lines from 3 to 5); second, it continually selects paths from \( Sp \) with the next criterion, and then replaces \( Sp \) with the newly selected paths (lines from 6 to 9). If there is only one path in \( Sp \), then it is the final path (line 12); otherwise, this procedure keeps selecting paths from \( Sp \) according to the next criterion in the order. If all the criteria in the priority order have been applied and \( Sp \) still includes multiple paths, then all the paths in \( Sp \) are regarded as the final paths. Ultimately, the set \( Sp \) includes all the semantic paths resulting from the priority order. This computation obtains either one path or several equally-optimal paths. Figure 4.10 presents the workflow of routing with priority ordering.
In practice, when considering the profile of users the importance of each type is sorted for distinct users with different profiles. These sequences of types of path are transformed into the priority ordering of criteria.

To sum up, in order to compute logical paths for a specific user, this research requires: 1) a user-related logical network; 2) a priority ordering of criteria to the user; and 3) the routing workflow. This workflow may generate one or several paths at the end of the computation. The final paths are regarded with the same importance. In an application, one can visualize all the paths and ask users to select one, or randomly choose one.

Besides the Lexicographical Goal Programming, there are more means to reduce the chance of getting multiple logical paths. For example, to design more criteria by assigning distinct weights in the logical network. Multiple logical paths can result from simple weights, such as HC nodes with 1 and the others with 0 in minimum HC. In this case, if VC nodes are set to a larger value (e.g., 3) to emphasize their importance, then the previously multiple paths may be reduced. However, the difficulty is to motivate the increased weights of VC, i.e., what kinds of scenarios need them. Thus this topic is left for future investigation.

§ 4.5 Summary

Routing on the logical network is related to user status (e.g., motion ability, size of user and access permission) and her/his preference for indoor spaces and paths. Two users with different access permissions would get distinct logical networks for the same building. In the same network, two users with different preferences may obtain different logical paths. This chapter answers and explains the following research sub-question:

4. What kind of routing criteria can be built (or specified) by using the semantics of indoor spaces?

Section 4.1 introduces the motivation of routing on the logical network, i.e., to simulate human wayfinding behaviours. In such a routing users involve different preferences of space. In order to reflect the preferences, routing criteria are designed based on the semantics of indoor spaces.

Section 4.2 presents the derivation of a logical network. Based on the association of NU and OPN in INSM, the logical network can be automatically derived. The semantics of nodes in the logical network is retrieved from INSM.

For the research question, Section 4.3 explains why the semantics of spaces is needed by pedestrians for routing and presents the advantages of using INSM semantics for routing. Then six routing criteria are proposed on the INSM semantics: 1) Minimum NavigableUnit minimizes the number of traversed spaces; 2) Minimum Horizontal-Connector minimizes the number of HorizontalConnector in a path; 3) Minimum VerticalUnit minimizes the number of VerticalUnit in a path; 4) Central HorizontalConnector preserves the high accumulated centrality of HorizontalConnector in a path; 5) HorizontalConnector Prior prioritizes HorizontalConnector nodes and also makes the path as
direct as possible to the target space; and 6) VerticalUnit Prior prioritizes VerticalUnit nodes and also makes the path as direct as possible to the target space. Besides, three existing pedestrian wayfinding strategies [CTG+06] are adopted as algorithms in Section 4.3, i.e., floor strategy, flat location strategy, and SOI strategy. These proposed criteria are not an exhaustive list. More routing criteria can be designed in terms of other user needs on space.

Section 4.4 introduces the computational steps of routing on the logical network that are weighted according to the semantics of logical nodes for different criteria. Logical paths can be computed for users by using only a singular criterion or a sequence of ordered criteria. A criterion reflects a specific user’s preference on indoor spaces; and the user profile can be linked to the priority ordering of multiple criteria for the user.

As mentioned in Section 3.4.1, two options (L1.1 without SOI and L1.2 with ordered SOI) are summarized for obtaining logical paths. Option L1.1 is related to the six proposed criteria, the floor and flat location strategies. These criteria and the two strategies can automatically derive logical paths for a user. Option L1.2 is reflected by the SOI strategy where a user needs to specify nodes (SOIs) to be passed in the logical network.

In the two-level routing approach, a logical path is computed first when geometric paths are required. The geometric network can be generated ‘on the fly’ for the relevant spaces after the logical path is given. Specifically, a new logical path is needed when there is no available geometric path in the geometric network. The next chapter will present the generation of the geometric network and routing in this network. Routing in the geometric network generates accessible and obstacle-avoiding paths for users with a given size.
This chapter presents routing on the geometric network of a building. As addressed in previous chapters, this routing aims to compute a detailed geometric path that avoids indoor obstacles with respect to pedestrian-related sizes. Indoor spaces can contain a large variety of static objects, which obstruct navigation, such as desks, chairs, tables and other furniture. A geometric network can be created for specified spaces by considering the shapes of the indoor static obstacles. In this thesis, a visibility graph method [OW88, AW88] is adopted to generate geometric networks regarding obstacles. In order to compute accessible paths for a given user, the distance between the obstacles and their distance to walls should be taken into account. In the 2D plane, a pedestrian-related size is defined as the diameter of the circumcircle that covers the pedestrian and the devices/objects (e.g., wheelchair or suitcase) she/he carries/drives. The size of pedestrian is named 'user size' in short.

This chapter focuses on the generation of geometric networks and proposes the generation approach: for the selected spaces of a given logical path, a subnetwork is created for each space and then the subnetworks form the geometric network for the selected spaces. Routing on the geometric network can be computed by using traditional methods, such as the Dijkstra algorithm [Dij59]. Section 5.1 presents the motivation for using the visibility graph method to construct geometric networks, which can preserve corner points (i.e., turns) in geometric paths. Section 5.2 elaborates the generation of geometric networks considering user sizes, including the method to include user size, creation of geometric edges, the network formed by the edges, and geometric network generation for changing user sizes. Section 5.3 briefly summarizes the whole chapter. This chapter is based on two author’s own publications: [LZ11a, LZ15].

§ 5.1 Motivation

As presented in Chapter 3, the geometric network is created on the second level in selected spaces. This section introduces the general process of geometric network derivation for indoor spaces with obstacles. Except for obstacles, a user needs to specify POIs and their ordering in routing option G2.2 (with ordered POI, see Section 3.4.2). In this case the generation of the geometric network needs to input the given ordered POIs.

Chapter 2 discussed several approaches for computing the geometric network. The commonly used S-MAT [EE99, CL09] paths do not provide direct paths inside of irregular indoor spaces (see Figure 5.1), and therefore other approaches have been investigated. For example, the approach direct path graph (DPG) [YS11] computes the shortest paths in spaces without creating a complete geometric network. However, this method cannot ensure the shortest path in a given space (see Figure 5.2), e.g., some irregular-shaped spaces [LZ11b]. For comparison, Figure 5.2 illustrates a DPG path between two doors in an irregular-shaped room. The DPG path is derived by dividing the straight line between the start and target locations each time when the line intersects
FIGURE 5.1  Comparison of S-MAT and direct paths. A S-MAT path (solid lines) along a corridor is D1-M1-M2-M3-D3, and the shorter direct path (dashed lines) is D1-S1-D3 without considering user size (from [LZ11a]).

a non-convex boundary. The path computed based on the VG (Figure 5.2b) is the genuine shortest path between the two locations.

FIGURE 5.2  Comparison of a DPG path and the shortest path. On the left is the DPG path, and on the right is the shortest path between the start and target locations (from [LZ11b]).

As mentioned in Chapter 2, this thesis adopts the visibility graph (VG) method [OW88, AW88] to facilitate the generation of geometric networks. Indoor obstacles significantly influence the form of a geometric network. This thesis concentrates on static obstacles represented by disjointed 2D polygons (e.g., floor plans) or 3D volumes (e.g., IFC data). An obstacle (i.e., an object) occupies a certain region in a space. To be able to create a geometric network, a number of vertices are considered: opening centers, obstacle vertices, POIs and non-convex corners of the related spaces. Edges of the VG are the direct path between two VG nodes.
This thesis denotes geometric paths by straight lines. It is assumed that a user can perceive a turn (or a corner) as a significant ‘landmark’. This path is computed considering the corner points and they are indicated in the path. In reality a user would keep a distance to obstacle corners as a collision-free path but such computation and visualization of the path is not realized. This thesis adopts the simple visualization of a path which clearly includes corner locations, which highlights the turns for pedestrian motions. However, the distance between obstacles is taken into account, which lays a foundation for routing regarding user sizes.

According to the definition in Chapter 3, a geometric network consists of the shortest paths among opening locations and POIs. Specifically, the geometric network is a subset of the VG (see Figure 5.3). Nodes of the VG include obstacle vertices, door center locations, POI and non-convex vertices of the space. Once the VG is created for a space, the geometric network for the space can be derived by computing all the shortest paths between openings (doors) and POIs (see Figure 5.3). Similarly, geometric networks can be created in a set of spaces or all indoor spaces together.

\[ \text{FIGURE 5.3 } \text{Comparison of VG and geometric networks. (a) VG. Dots represent VG nodes and black lines are VG edges. Blue polygons are obstacles; and (b) The geometric network. Dots represent geometric nodes (doors), and the shortest paths among the nodes are geometric edges.} \]

§ 5.2 Geometric network generation

A geometric network can be derived from the geometry stored in the INSM. In this chapter, nodes are referred to as the center of door locations, and these nodes normally would not change in the building. It should be noted that nodes of a geometric network are always the same, but edges may differ for distinct users. When a new geometric network is created for a given user, only the edges need to be updated among these doors according to the user. Thus this section focuses on the computation of geometric edges among door nodes considering user sizes. In this thesis the term ‘bottleneck’ is introduced to indicate the region (i.e., inaccessible gap) between two obstacles where a user with the given dimension cannot pass through.

Subsection 5.2.1 presents the adopted method to consider user size for routing in geometric networks. This method groups obstacles by comparing distances among obstacles to a user’s size, and then derives geometric edges among these groups of obstacles. Subsection 5.2.2 briefly introduces the computation of accessible geometric edges between two doors, which includes six steps: 1) find bottlenecks (Subsection
§ 5.2.1 Approach to consider user size

User sizes are specifically important for facility management and maintenance. For example, a member of the maintenance staff in a factory operates a large vehicle (see Figure 5.4a, or a staff member in an airport transports different goods with a wheeled cart (see Figure 5.4b). All of these cases require paths that can consider the total size of pedestrians and the objects they operate/carry.

As discussed in Chapter 2, robot motion planning has always taken into consideration robot dimensions. The so-called Minkowski sum method [dBCvKO08, Coe12] has been commonly applied to identify inaccessible areas for a robot, yet the Minkowski sum approach tends to generate non-simple geometry (see Figure 5.5). Such a geometry is not convenient for creating a navigation network, because it can derive inner rings and self-intersection of polygons. Thus the generated geometry needs to be merged and cleaned. Edges touch each other at the polygon of the merged Minkowski sums, which increases the redundancy of vertices (Figure 5.5b). Furthermore, the generated polygons (see Figure 5.5) either include arcs or too many vertices (e.g., the ‘curved’ parts). Figure 5.5 presents two union results of the Minkowski sums. An isolated inner ring (see Figure 5.5a) is part of the merged Minkowski sums of the six objects. The inner ring can be removed since the user can never access it from outside. Figure 5.5b presents the case of self-intersection: several edges touch each other or overlap within a tolerance at the polygon of the merged Minkowski sums, which increases the redundancy of vertices. They need to be cleaned, i.e., the edges are merged and removed. The
generated polygons include arcs or too many vertices (e.g., the ‘curved’ parts), which complicates the creation of the network.

![Generated polygons](Image)

**FIGURE 5.5** The union of Minkowski sums contains inner rings and self-intersections. (a) The union of Minkowski sums with one inner ring (the circle denotes a user); and (b) Self-intersection and inner rings of the Minkowski sums.

This thesis addresses the inner rings and self-intersection issues by directly measuring the *minimum distance (MD)* between the obstacles and determining the boundaries of obstacle-occupied areas with fewer vertices. One can find bottlenecks between obstacles with MDs. A bottleneck is inaccessible since the MD of the bottleneck is shorter than the user size. Indoor obstacles sharing bottlenecks are regarded as a group. A simple polygon is computed to bound multiple obstacles when their inside is not accessible to the user. The term ‘simple’ refers to the simple features in the standard [OGC11] of the Open Geospatial Consortium. With the same objects in Figure 5.5, simple polygons are derived as the boundaries of the two groups of obstacles (see Figure 5.6). Compared to Figure 5.5a, the polygon (i.e., boundary) in Figure 5.6a has no inner rings. The polygon in Figure 5.6b has fewer vertices compared to the one in Figure 5.5b.

An accessible shortest path (i.e., geometric edge) can be computed between two given locations with groups of obstacles when the obstacle groups influence the path. Regarding this approach, all the following illustrations are only about ‘door-to-door’ computation in a space. In this section, only doors are considered to compute geometric edges, and no POIs or windows are presented.

In Figure 5.7a, the obstacles on the left side of (outside the convex hull denoted by black lines) the room do not influence the direct path between the two doors (the blue line) based on the VG. These obstacles are excluded by the method in Section 5.2.2 about selecting obstacle groups with a convex hull with the proof in Appendix A. Figure 5.7 illustrates geometric edge generation with simple boundaries of obstacle groups for a user:

1. A VG for the user is constructed with only the two groups on the right side of the room (see Figure 5.7b);
2. To consider the gaps between walls and obstacles, a buffer is introduced (see Figure 5.7b) which is equidistant from all of the walls with the user size. The final VG for the user is computed by considering the inaccessible gaps in the area of the buffer. In this buffer, the inaccessible gaps between the walls and the obstacles are presented as red lines in Figure 5.7b. The edges of the VG that intersect these inaccessible gaps are removed.

3. Subsequently, the shortest path \( \text{(i.e., geometric edge)} \) is computed on the VG (see Figure 5.7c). The path on the nodes of the boundaries (see Figure 5.7c) is a schematic path, and it provides clear directions and illustrates where the user can pass.

4. A realistic path considering the size of the user is visualized in Figure 5.7e.

For the sake of simplicity, the remainder of this thesis only visualizes paths as schematic paths. But as an example, a simple method to reconstruct a realistic path is presented in Figure 5.7d:

1. Find the related obstacle groups to the schematic path;
2. Compute Minkowski sums for these groups;
3. Locate the intersection points of these Minkowski sums and the schematic path, and find the part of the schematic path inside the Minkowski sums;
4. If any part of the schematic path is inside these Minkowski sums, then replace the part by the bound of the corresponding Minkowski sum, which forms a realistic path as a user’s accurate trail (see Figure 5.7e).

For a space, a geometric network consists of all the accessible paths when a user size needs to be considered. As mentioned before, based on a VG the shortest paths among door locations and/or POIs form a geometric network. For distinct user sizes, different geometric networks could be created in a space.
FIGURE 5.7  Path computation with simple boundaries of obstacle groups for a user: (a) Selection of groups of obstacles between two locations. The circle denotes the user; (b) the VG considering the inaccessible gaps between obstacles and walls; (c) a schematic representation of the computed ‘shortest’ path (in black) in the VG; (d) a way to compute the realistic path; and (e) a realistic path by considering the user size.
§ 5.2.2 Edge generation of geometric networks

In this thesis, a new path computation approach is proposed to avoid static obstacles and obtain indoor paths with respect to the user dimension. In a space, the approach computes the accessible shortest paths between two locations for users with different sizes. A geometric network is created for a set of spaces, which is the union of the sub-networks in each space.

The proposed approach excludes inaccessible spaces for a user in six steps (see the highlighted parts in Figure 5.8):

1. Compute the MDs between the indoor obstacles of each room by applying the Rotation Callipers algorithm [Tou83]. Inaccessible gaps are obtained when an MD between obstacles is shorter than the user’s dimensions (Section 5.2.3).

2. Group obstacles according to the inaccessible gaps (Section 5.2.4).

3. Between two locations, select necessary groups of obstacles to construct a geometric network by applying a convex hull-based method (which will be elaborated in Section 5.2.5).

4. Compute non-overlapping boundaries with simple geometry for the selected groups based on the Delaunay Triangulation [dBCvKO08] and the alpha shape method [EM94, AEF+95] (Section 5.2.6).

5. Compute MDs between these selected groups and walls, and locate inaccessible gaps (Section 5.2.7).

6. For single spaces, create geometric edges for users with the groups’ boundaries and remove edges crossing inaccessible gaps between the boundaries and walls (Section 5.2.8). Then form a geometric network for related spaces with these geometric edges (Section 5.2.9).

Finally, a path can be computed to accommodate the user with the given size. The following subsections elaborate each of the above steps.

FIGURE 5.8 Overview of the proposed approach.
§ 5.2.3 Compute MD between obstacles and find inaccessible gaps

As mentioned before, the proposed approach makes use of the MD between obstacles, which aims to detect bottlenecks. Obstacles with bottlenecks are to be categorized into a group. A path around the group is provided to the user. This paper employed the rotation callipers algorithm [Tou83] to compute the MD between obstacles. As this algorithm is applied to convex polygons, non-convex obstacles are subdivided into convex parts to compute their MDs with other obstacles. Figure 5.9 illustrates the MD computation between two non-convex polygons:

1. The two polygons are subdivided into different convex parts [CD85] (Figure 5.9a), such as convex parts 1 and 2 to the non-convex polygon A, and the convex parts 3 and 4 to polygon B;

2. To compute the MD between each convex part of A and that of B with the rotation callipers algorithm, and four MDs are derived i.e., $D_{13}$, $D_{23}$, $D_{14}$, and $D_{24}$ (Figure 5.9b). The subscripts denote the related convex parts of A and B;

3. Finally, the lowest MD is the MD between the non-convex polygons A and B (Figure 5.9c).

The MDs of each static obstacle are computed with the other obstacles. If an MD is smaller than a given dimension, then it indicates a bottleneck (i.e., the user with the given size cannot pass). In this manner, all of the bottlenecks between the obstacles are collected (see Figure 5.10).

FIGURE 5.9 MD computation for two non-convex polygons. (a) Each non-convex polygon becomes different convex parts; (b) MDs between each convex part of A and that of B, i.e., $D_{13}$, $D_{23}$, $D_{14}$, and $D_{24}$ (in red); and (c) The lowest MD is the MD between A and B, i.e., $MD_{AB}$ is $D_{13}$.

FIGURE 5.10 Bottleneck detection between obstacles. The thick lines indicate the bottlenecks.
Figure 5.10 presents bottlenecks between obstacles for the user with a given size. The bottlenecks are denoted with the lines linking the convex hulls (CH for short) of obstacles (see Figure 5.10). If an obstacle has no bottleneck with the other obstacles, then the obstacle is an individual group (e.g., Obstacle 11). Otherwise, the obstacles that ‘connect’ to each other by these lines are put into the same group (e.g., Obstacles 1–4).

§ 5.2.4 Group obstacles

This section introduces the method for grouping obstacles by using the MD between them. Obstacles with bottlenecks (i.e., the MDs are smaller than a user’s size) are put into a group. A linked list is created for each obstacle. Each bottleneck between two obstacles is regarded as a connection of them. The linked list contains all of the other connected obstacles. The following steps illustrate the process (see Figure 5.11).

- Step 1. Pick an unchecked obstacle as the current obstacle ‘obs’. If there is no unchecked obstacle, then go to Step 5. Otherwise, create an empty group ‘gop’; add obs to gop; and go to Step 2.

- Step 2. In the linked list of obs, add all of the unchecked members within the distance to gop.

- Step 3. For the previously added members, add all of the unchecked members in their linked lists to the gop.

- Step 4. Repeat Step 3 until no unchecked members are found. Then, the obstacles in the gop form a group. Go to Step 1.

- Step 5. All of the groups have been identified. Count the number of groups and assign each group an ID.

Figure 5.11 presents an example of grouping obstacles. The groups of obstacles are the actual obstacles for users with a given size. The user has to avoid any obstacle groups that can have different shapes. Accordingly, the boundary of every obstacle group needs to be generated, which will be addressed in Subsection 5.2.6.

§ 5.2.5 Select obstacle groups

This step is introduced to reduce the number of groups that will be used for path computation. Between two locations, some indoor objects may not interfere with the path for a user with a given size. Hence, it is necessary to ascertain the groups of obstacles that influence a path.

In a room, the shortest path is computed first between two locations without respect to the obstacles. The path is named the direct path. The direct path and CH of obstacles are employed to select obstacle groups. The selection process consists of the following steps:

- Step 1. Find the obstacles intersecting the direct path.
FIGURE 5.11  Grouping 11 obstacles into three groups with respect to bottlenecks.

- Step 2. Select all of the obstacles from the groups that have an obstacle intersecting the direct path.

- Step 3. Compute a CH with the nodes of all of the selected obstacles and the direct path.

- Step 4. If the current CH intersects or contains new obstacles, look up the groups of the new obstacles, and select all of the obstacles from the new groups. Re-compute a CH.

- Step 5. Iterate Step 4 until there are no other obstacles included by the current CH.

The obstacle selection aims to choose obstacle groups in a limited region for path finding. The iterative procedure will stop when the final CH (FCH for short) does not include any new obstacle. For a user with a given size, the shortest path in the VG can be found in the FCH or the bound of the FCH (see Lemma 2 and 3 in Appendix A).

To ensure the correctness of the above algorithm, three Lemmas are introduced whose proof can be found in Appendix A:

- **Lemma 1** If a polygon contains some indoor static obstacles, then the polygon also contains all visibility edges of the VG derived with these obstacles.

- **Lemma 2** Given a user size, related grouped obstacles, and the start and target in a space, if a computed FCH does not intersect exterior obstacles, then the shortest path from the start to the target is either inside or in the bound of the FCH.

- **Lemma 3** Given a user size, related grouped obstacles, and the start and target in a space, the real shortest path from the start to the target is the shortest path derived in the VG of the obstacles in the FCH.
Lemma 1 indicates a CH (polygon) contains the whole VG among the obstacles in the CH. Lemma 1 is the base for Lemma 2; Lemma 2 ensures the shortest path inside or in the bound of the FCH; and Lemma 3 ensures that the shortest path derived from the FCH is the real shortest path among all the obstacles for the given size of user.

Figure 5.12 illustrates the selection process. The obstacles have been grouped according to a given dimension of users. A direct path is computed between two locations. First, the direct path intersects two obstacles (Step 1). The two obstacles belong to two different groups; second, all of the obstacles in the two groups are selected (Step 2); third, a CH is computed with the selected obstacles (Step 3). Then, the CH intersects two new obstacles; fourth, the group of the two new obstacles is found. All of the obstacles in the group are selected, and then, a new CH is computed (Step 4); finally, the new CH has no other intersections (Step 5). Thus, the CH is the FCH, and the three groups of obstacles are selected. The extreme case is that all of the obstacles in a room are selected in the FCH. This means that all of the obstacles will be used for path computation. This case may happen in an obstacle-dense scenario. Large sizes of humans and equipment can also result in the extreme case. But in common indoor scenarios only some of the obstacles in a room are selected, especially in a large room/hall where a user just needs to pass through part of the room from the start to the target location.
§ 5.2.6  Create boundaries for selected groups

This section presents the generation of non-overlapping boundaries for the selected obstacle groups. The alpha shapes [EM94, AEF+ 95] method can be employed to compute non-convex boundaries of a point set. In the 2D plane, the alpha shapes of a point set are different polygons formed by this point set. Each alpha shape (i.e., a polygon) is determined by a value (i.e., the alpha value). The alpha shape is the CH when the alpha value approaches infinity, and it becomes the set of points when the alpha value equals zero [EM94]. Other alpha values in between zero and infinity correspond to a number of non-convex polygons of the point set. The alpha shape method is adopted to compute non-convex boundaries for obstacle groups, which ensures separate boundaries for different groups of obstacles. Thus the boundary of an obstacle group is an alpha shape of the vertices of all the obstacles in the group.

A common method [EM94] is used to compute alpha shapes by employing Delaunay Triangulation (DT) [dBCvKO08]. For the set of points $P$, the DT is a decomposition consisting of a set of triangles in which there is no other point inside the circumcircle of each triangle [dBCvKO08]. Based on the DT, the alpha test results in the alpha shape by using an alpha value: for each triangle in the DT, if the length of every edge in the triangle is shorter than the alpha value, then the triangle needs to be preserved; otherwise, the triangle will be removed. The boundary of the union of all the preserved triangles forms the alpha shape.

The alpha shape is computed based on the alpha value equal to the given size of a user. As a result, there is only one alpha shape (i.e., the boundary) for the user. This method generates non-overlapping boundaries of all the obstacle groups (see Figure 5.13). The procedure for boundary generation is presented below:

- Step 1. For a selected obstacle group, check the number of obstacles. If the group includes only one obstacle, then the obstacle’s polygon is the boundary. Otherwise, go on to Step 2.
Step 2. Create a DT with the vertices of all the obstacles in the group, and assign the given size of the user to the alpha value.

Step 3. Compute all of the lengths of the edges of each triangle in the DT. Preserve a triangle if its edges' lengths are all smaller than the alpha value.

Step 4. In all of the preserved triangles, find the edges only used for one triangle (i.e., other edges shared with two triangles are interior edges). Form the edges into a boundary.

Three groups of obstacles are shown in Figure 5.14. The boundaries are derived in the same number of obstacle groups. The alpha value is set to the user size 0.8 meter (m). At first, two groups' DTs overlap. After the alpha test has been applied to the DTs, non-overlapping boundaries are computed for the groups.

![Figure 5.14 Boundary generation of three obstacle groups for a user with size 0.8m.](image)

§ 5.2.7 Compute the MD between the boundaries of the obstacle group and walls

There are two possibilities for paths. Paths can be found either in the gaps between the groups of obstacles or the gaps between obstacles and walls of a space. If such a path is not found, this space cannot be passed by a user with this size.

Similar to the bottlenecks between obstacles, bottlenecks are also defined for a wall as the inaccessible gap between the wall and the boundary of an obstacle group. For a user with a given size, bottlenecks are detected for walls by using the buffer of a space (see Figure 5.15). The buffer is computed with the offset value equal to the user's width. In the selected obstacle groups, if a node of the boundary is inside the buffer, then a perpendicular line is derived for the wall (see Figure 5.15). The perpendicular line represents the bottleneck (i.e., not possible to pass through) for the wall.

§ 5.2.8 Create a VG considering inaccessible gaps with walls

For a user with a given size, a VG can be created in a space with the boundaries of the selected obstacle groups. The edges crossing the bottlenecks to the walls are inaccessible to the user (see Figure 5.15). The edges should be removed for path computation.
The VG is adopted to compute geometric edges. Figure 5.16 illustrates the creation of a VG and presents the shortest path between two locations for users of 0.8 m. First, the VG is created with the nodes of three selected group boundaries (see Figure 5.16a). Second, a buffer of the space is computed, and the three boundaries overlap the buffer (see Figure 5.16b). Third, all of the bottlenecks between the boundaries and walls are found (see Figure 5.16c). Consequently, the inaccessible visibility edges are located. Fourth, the inaccessible visibility edges are removed (see Figure 5.16d). Finally, the shortest path (see Figure 5.16e) is computed in the VG. A user with the given size can follow the path. As addressed in Section 5.1, though the computed path touches corners of obstacles (Figure 5.16e), the schematic visualization of geometric edges is used instead of precise trails for a user, i.e., the form derived from the above VG.
FIGURE 5.16  A visibility graph (VG) and a space buffer. Inaccessible edges are removed and a path is computed for users with size 0.8m. (a) The VG; (b) Space buffer; (c) Inaccessible edges in the bottlenecks; (d) The VG without inaccessible edges; and (e) The computed geometric edge.
§ 5.2.9 Network from edges

The above approach is used to generate geometric edges for users with different sizes. For each given user size the proposed approach is employed to compute all of the shortest paths among the doors/POIs in a space (see Figure 5.19). These shortest paths (e.g., geometric edges) form the geometric network for the given user.

A geometric network can be derived in a space. Given a user size, the proposed approach is iterated for every pair of start and target doors/POIs in a space:

1. For all obstacles, conduct the first two steps ‘Compute MD between obstacles and find inaccessible gaps’ and ‘Group obstacles’.

2. Iterate the third step ‘Select obstacle groups’ for each pair of start and target, and record all the groups selected with each pair of start and target. Then the union of all these selected obstacle groups is used to compute the geometric network in the space.

3. Apply the fourth step ‘Create boundaries for selected groups’ for the above obstacle groups.

4. On the above obstacle groups, conduct the fifth step ‘Compute the MD between the boundaries of the obstacle group and walls’.

5. Derive a VG for routing by the final step ‘Create a VG considering inaccessible gaps with walls’.

6. In this VG compute all the shortest paths among all the pairs of doors/POIs in the space for the given user size. These paths form the geometric network.

Obstacle groups selected with each pair of start and target is the subset of all the obstacle groups. According to Lemma 3 (see Appendix A), given a start and a target and a user size, the shortest path (i.e., geometric edge) can be computed with the selected obstacle groups. Therefore, the above process can ensure geometric edges for the geometric network in this space.

In the above process, unrelated obstacle groups are excluded for VG generation by the iteration of ‘Select obstacle groups’ for each pair of start and target. But in some cases (e.g., very large user sizes and a few obstacle groups), the iteration may select all the obstacle groups in the space. In such cases, the iteration would stop, and immediately go to the next step (‘Create boundaries for selected groups’) with all the obstacle groups in the space.

Based on the derived VG, all the geometric edges between the doors can be computed in this space. Figure 5.17 gives an example of geometric networks for a space. Figure 5.17a presents a geometric network between doors for a user with size 0.6m. In Figure 5.17b, two POIs (in red) are specified and added by the user and another geometric network is computed that consists of edges among the doors and POIs. One can find that the two POIs significantly change the geometric network regarding only doors (Figure 5.17a). Once the POI changes, the geometric network needs to be regenerated. Therefore, it is not necessary to store geometric networks when POIs are frequently added and/or deleted.
The proposed approach is also used to form a geometric network for a number of spaces. As addressed before, this research focuses on navigational functions of indoor spaces, i.e., to find indoor paths by considering different space choices. In this sense, routing criteria for the selection of spaces (see Chapter 4) are prior to the geometric criteria (i.e., the shortest distance). Therefore, in this thesis a geometric network is the combination of all the subnetworks in the given spaces.

Two spaces are connected via specific doors. According to the sequence of spaces, the sequence of doors can be confirmed. Then in each space the geometric network is generated according to the related doors. For example, Figure 5.18 presents a geometric network for the user size 0.6m in a sequence of spaces. Figure 5.18a shows the space sequence is S1-S2-S3-S4. The highlighted doors are geometric nodes. The sequence of doors is D1-D2-[D3,D4]-D5 which depends on the space sequence. In the space S2, there are two geometric edges from D2: D2-D3 and D2-D4; in the space S3, there are two geometric edges to D5, i.e., D3-D5 and D4-D5. To construct this geometric network, five obstacle groups are selected in S1 and S2 (see Figure 5.18b). The geometric edges between doors for the given size can be stored to be reused when the same space sequence is frequently used for logical paths.

§ 5.2.10 Geometric network for users with changing sizes

The computed geometric networks contain only edges that can be followed by a specific user. As mentioned before, doors for the geometric networks are the same. However, if there is not any accessible path starting from or leading to a door node, then this node can be removed from the geometric network since it is isolated.
Regarding the same set of geometric nodes (doors) in a space, Figure 5.19 illustrates different edges for distinct user sizes. Figure 5.19a presents the VG created with all the obstacles without a user size. It is visible that VG has too many edges, which will influence the computational performance. The geometric networks (Figure 5.19b–d) consist of the shortest paths among the doors, and they are computed for the users with the sizes 0.5 m, 0.6 m, 0.8 m, respectively. For other users with the given sizes, geometric networks can be immediately created in the space. Moreover, the inaccessibility among doors can be recorded and be reported to the users. For instance, there is no geometric edge between the doors $D_2$ and $D_8$ for users of 0.8m (Figure 5.19d), but geometric edges between the two doors exist for users of 0.5m and 0.6m (Figure 5.19b-c).
FIGURE 5.19  Geometric networks for different user sizes. (a) A complete VG regardless of a user’s size; (b) The geometric network for size 0.5 m; (c) The geometric network for size 0.6 m; and (d) The geometric network for size 0.8 m.
Generally the geometric network of the larger size can be re-used in those of smaller sizes. For example, the geometric network of 0.8m can be computed and stored, and then edges of this network can be added to the geometric network of 0.6m or 0.5m.

The geometric networks can be maintained in a database and used according to path requests. The user can also be informed if no path exists. The accessibility information can even be recorded as the attribute to each space and employed for users to estimate generally the possibilities to pass through certain spaces.

§ 5.3 Summary

This chapter elaborates on the generation of user-dedicated geometric networks, taking into account the size of the user. This chapter responds to the following research sub-question:

5. Which approach should be used to compute the exact geometric description of accessible paths according to the size of a user?

Accessible paths are computed on the geometric network for a given user. The generation of a geometric network is based on the Visibility Graph (VG) method [OW88, AW88], which is motivated in Section 5.1. Section 5.2 introduced the new method to derive accessible geometric edges for a user in a single space to avoid obstacles. This method includes the following steps: 1) compute the minimum distance (MD) between obstacles; 2) group obstacles according to inaccessible gaps; 3) select obstacle groups with the Convex Hull method; 4) create boundaries for selected groups; 5) compute the MD between obstacles and walls; and 6) create the VG considering all the inaccessible gaps and compute the geometric edge. All the geometric edges in a space form the geometric network for this space. The generation of a geometric network for a sequence of spaces is also illustrated.

Besides doors, this approach can be used to include POIs or other openings (e.g., windows) in the geometric network. The geometric network should be re-created before routing, when a user specifies a new POI or multiple ordered POIs. In this case, the network cannot be pre-stored since the POI was not known in advance.

Section 5.2 also presents the example of geometric networks for changing user sizes. These geometric networks contain the same set of geometric nodes but distinct edges. POI can be changed according to different users, but doors of a building are constant. Thus the edges between doors can be stored, and edges related to POI can be computed in real time. As the illustrated geometric networks are only in regard to doors, these networks can be pre-computed and stored for a user or computed on demand. For instance, the geometric network can also be generated when the user walks into the space.

To compute accessible paths between two locations at different spaces, a geometric network is created regarding a set of related spaces which can result from routing on the logical network. So far, routing on the logical and the geometric network has been
thoroughly introduced. Chapter 6 and 7 will present the realization of the two-level routing approach.
6 Realization of one-level routing

Chapter 3 has introduced the design of the two-level routing approach, such as the definitions of logical (abstract level) and geometric (detailed level) networks, the INSM data model that structures building data for routing, the logical and geometric networks and routing options using the two types of network. Chapter 4 and 5 have elaborated the routing procedures and algorithms on the logical and the geometric networks, respectively.

This chapter presents an assessment based on the realization of the proposed concepts. Section 6.1 introduces the adopted software tools, building data and user-related information in a profile. Section 6.2 introduces the generation of the INSM from the used datasets, including the mapping procedure of the INSM from photographic or digital floor plans, IFC and CityGML LoD4 models. Section 6.3 presents tests of the routing on the logical and the geometric networks of buildings independently, which are named logical routing and geometric routing. Either of them uses just one type of network for routing, thus logical routing and geometric routing are regarded as one-level routing. Chapter 7 will address integrated two-level routing. Tests of logical routing illustrate the implementation of three typical criteria: Floor strategy, Minimum NavigableUnit (NU) and Central HorizontalConnector (HC); tests of geometric routing show the following results: 1) for different users; 2) the same user with changing sizes; and 3) the generation of a large geometric network from the ones in each space considering POIs and different user sizes.

Section 6.4 discusses the test results, explains the main use of INSMs, and indicates possible extensions of logical routing and geometric routing. This chapter closes with some concluding remarks in Section 6.5.

§ 6.1 Software, data, and user profiles

The software and tools used in this research include MicroStation V8i of Bentley Systems [Sys16c], Visual Studio 10.0 and igraph (an open-sourced Python package) [ict16] for graph computation and visualization. MicroStation is Computer Aided Design (CAD) software which makes it possible to digitalize photographic building data and to visualize routing results. MicroStation provides SDK for secondary development to customize functions. The NativeCode method is used, i.e., to compile C or C++ source files with MicroStation/Windows resource files and then generate executable applications. The NativeCode method has a better performance in computation-intensive tasks compared to MicroStation BASIC applications, and it can make use of debugging functionalities in the Microsoft Visual Studio 10.0 environment.

One desktop application with integrated functionalities (e.g., network creation and routing) is developed in MicroStation. It supports the creation of a logical network, helps to derive geometric networks ‘on the fly’, and conducts logical and geometric routing according to user information and preferences (e.g., SpaceOfInterests and PointOfInterests). Another application developed with the igraph package is used to compute
and visualize logical paths. Logical paths are also visualized in the desktop application as a sequence of spaces. Geometric paths are computed and visualized as polylines in selected spaces in the MicroStation application.

The general steps are presented as follows:

- First the INSM of a building is created in MircoStation V8i;
- Within Visual Studio 10.0, logical networks are created and are enriched with the INSM semantics;
- Logical paths are computed for users and visualized with igraph;
- Geometric networks are created for users, and geometric paths are computed and visualized in MircoStation V8i.

As mentioned before, the Dijkstra algorithm [Dij59] is adopted for routing on logical and geometric networks. Three groups of routing test are performed:

- Routing just on the logical network, i.e., logical routing. For routing with multiple criteria, the computation adopts the workflow of routing with a priority ordering of criteria (see Chapter 4). In the following experiments, the priority order of criteria was decided according to specification in user profiles.
- Routing just on the geometric network, i.e., geometric routing. To create a geometric network for a specific user, steps presented in Chapter 5 were implemented including computing minimum distances (MD), grouping obstacles, creating group boundaries and creating VGs. Full steps were implemented by C++ language in the Visual Studio 10.0 environment.
- Two-level routing, based on both logical and geometric networks. This is tested in the desktop and a mobile application which will be elaborated in Chapter 7.

§ 6.1.1 Data Preparation

This chapter presents tests in the desktop application for the assessment of one-level routing (i.e., to use one single type of network for routing). The first step is data preparation. The input data is INSMs from which logical and geometric networks can be created automatically. As mentioned before, INSM is a data model for storage and management of indoor spaces whose geometry can be denoted by 3D volumes or surfaces. In this thesis, an INSM is referred to as 3D floor plans with the INSM semantics stored in different MicroStation layers. The geometry of indoor spaces is represented by 3D surfaces (e.g., stairs extended in vertical directions). Specifically, horizontal spaces are abstracted as flat polygons (in 3D space but without vertical walls) and vertical passages are symbolically represented by 3D polygons (reflecting boundaries).

In reality, datasets organized according to an INSM are not directly available. The most commonly available data source of buildings is floor plans (e.g., images of evacuation plans or architectural plans) in paper-based or digital form. 3D building models, such as BIM IFC and CityGML LoD4, include rich geometric details. They can be used for
indoor navigation. This chapter presents the mapping from such data sources to an INSM. Three types of data source are used:

1. no vector geometry and semantics of spaces (i.e., photographic floor plans);
2. no dedicated geometry for indoor spaces (i.e., architecture plans);
3. with 3D geometry and semantics of indoor spaces (i.e., IFC and CityGML LoD4), but no navigational semantics.

For the first type, the spaces and attached semantics are created manually. In the second type, digital architecture plans contain many lines and line strings to represent building elements but most of them are not closed. Thus, these spaces are manually closed and the original semantics is kept, such as Wall, Space, Door, Window, Stair, and so on. Work [Wu15] related to this thesis presents semi-automation for reconstructing 3D building models from architectural plans, which should comply with certain rules. More research by Dominguez-Martin et al. [DMvOFH + 15] presents the automation of building reconstruction with geometry at a non-refined level. This reconstruction result still cannot be as detailed as a complete geometric representation of the building. Besides, it does not provide semantics for indoor navigation. Therefore, it is difficult to automatically generate either 3D models or just 2D topologically clean floor plans (e.g., polygons with topology and semantics) from architectural plans.

The ideal group of input data is IFC and CityGML LoD4 models. The geometry of horizontal spaces in IFC is solid, but they are not directly used. Instead, only the accessible surface (e.g., ground surfaces) of a solid is used. Some vertical passages such as stairs are represented by a composite of solids (steps), which impedes the automatic abstraction of walkable surfaces. The floor surfaces of horizontal spaces are automatically abstracted, but surfaces representing vertical passages are manually created. For a CityGML model, spaces are also denoted by surfaces, which can be easily extracted. But the geometry of spaces is extended manually to create space semantics.

Tests of logical routing and geometric routing are implemented based on four photographic floor plans, one digital architecture plan, one IFC building model and one CityGML LoD4 model.
Photographic floor plans (the first type of data) are listed as follows:

- The image of Schiphol Airport, the Netherlands (Figure 6.1a). This image is downloaded in the webpage of the airport [Ams15]. It contains Stairs, Elevators and Escalators.

- The images of the 22, 23 and 24 floors of the Vermeertoren building, Delft, the Netherlands (Figure 6.1b). It contains Stairs and Elevators. The three floors share the same configuration and they are represented by the same image (Figure 6.1b). The image is accessed at: http://www.vermeertoren.nl/media/Vestia/nl_NL/Documents/Production_documents/Binnenwerk.pdf

- The image of the first floor of the Museum of Fine Arts (MFA), Boston, USA (Figure 6.1c). Stairs and Elevators related to this floor are included but not used, because this plan is used to test routing on only one floor. The image is accessed at: http://www.mfa.org/visit/plan-your-visit/floorplans.

- The image of the floor plan of a conventional neonatal intensive care unit (CNICU) at Sanford Children’s Hospital, USA (Figure 6.1d). Stairs and Elevators are not included because it is only for tests on this floor. This image is from the literature [SAKM+07].

The floor plans of Schiphol Airport, which is adopted as a complex building, include large irregular shaped spaces with many obstacles (see 6.1a). The building of the Vermeertoren in Delft contains many doors and it is a typical residential building. The first floor of the MFA includes many spaces and many of them are passages (see Figure 6.1c), which is a typical case to test multiple paths between two spaces on a floor. The CNICU contains many obstacles (see Figure 6.1d), so it is suitable for testing the proposed approach that computes geometric paths regarding user sizes among doors/POIs in single spaces.

A digital architecture plan (the second type of data) is listed as follows:

- The architecture plan of the ground floor of the Architecture Faculty building ‘BK’ at Delft University of Technology, the Netherlands (Figure 6.1e). It is a vector-based plan provided as a CAD file, and the data is authorized by the Map Room of this faculty. This plan is used to test routing on a single floor, and Stairs and Elevators are not included.

The ground floor of the BK building includes considerable indoor spaces and some of them contain a number of obstacles (see Figure 6.1e). The lines and symbols of this architecture plan are converted into spaces to analyse the influence of user sizes on geometric paths, and to discuss the creation of geometric networks with respect to these spaces.

Two 3D semantically rich models (the third type of data) are used:

- An IFC design model of a residential building (Figure 6.1f) which is provided by Bentley System Inc.;

- A CityGML LoD4 model of the old OTB building, Delft (Figure 6.1g) which is provided by the OTB Department of Architecture Faculty, Delft University of Technology.
6 Realization of one-level routing
FIGURE 6.1 Images of input datasets and the corresponding INSM representations. (a) Schiphol Airport; (b) The Vermeertoren building; (c) The MFA museum; (d) The CNICU floor plan; (e) The ground floor of the BK building (raw plan and clean version for routing); (f) A residential building provided by Bentley System Inc.; and (g) The old OTB building.
The two datasets are selected mainly based on the following reasons: 1) the indoor spaces of the buildings are well defined; 2) VerticalUnit (VU) information is easy to obtain. Subclasses of VU (Stair, Escalator and Elevator) can be distinguished with corresponding classes of the IFC and CityGML formats.

The IFC model of the residential building is selected as a simple case that contains a few spaces and only one staircase (Figure 6.1f). As introduced in Chapter 2, the IFC 2.3x standard includes several hundred classes. With the residential model, only the basic IFC classes are needed to support the mapping to the INSM. Furthermore, the data amount of the building geometry is small, which ensures the building geometry can be loaded with logical and geometric networks into a mobile application in the later test (Chapter 7).

The CityGML LoD4 model of the old OTB building (see Figure 6.1g) is adopted because it presents a typical office building that contains regular shaped offices and few obstacles. Each floor of the old OTB building has a similar configuration. The building contains one elevator and two stairs to change floors.

In the next part, a short name is given to all the above datasets: BK for the ground floor of the BK building at the Delft campus; Schiphol Airport for Schiphol Airport; Vermeertoren for the three floors of the Vermeertoren; MFA for the first floor of the MFA museum; CNICU for the CNICU at Sanford Children’s Hospital; Residence for the residential house; OTB for the old OTB building.

§ 6.1.2 User Profile

To reflect the relationship between routing and users, user profiles are introduced and connected to logical routing and geometric routing. As this research does not aim to develop a comprehensive user profile, a simplified version of a user profile is used, including four essential parameters: age, mobility, role and size of a user. More parameters and a comprehensive description about the user profiles for indoor navigation can be found in the work of Heckmann et al. [HSB+05] and Tsetsos et al. [TAKH06].

It is assumed that users have distinct routing criteria for logical paths. The parameter age makes it possible to distinguish suitable VUs for different users. A child may not be allowed to take the Escalator alone, and an elderly person may tend to use the Elevator to save effort. The mobility also helps to decide on proper VUs for a user. For example, a With-Wheel-Devices user can be a wheelchair user or a member of the maintenance staff with a vehicle equipment. They both prefer the Elevator to switch floors. Another example is an Effort-Saving-Motion user who prefers the Elevator/Escalator rather than the Stair, because she/he does not plan to walk vertically. The parameter role of a user indicates the user’s level of access permission. A Visitor can only visit public spaces, a Maintenance-Worker can access facility spaces (e.g., the electrical distribution room), but an Administrator can access all the indoor spaces of the building. The last parameter is the size of a user which reflects the dimension of the user. Values for these parameters considered in this thesis are listed below:

1. Age. Options: Child, Adult, Elderly Person.

4. Size. Options: values from a user.

<table>
<thead>
<tr>
<th>Age</th>
<th>Mobility With -Wheeled -Devices</th>
<th>Effort -Saving Motion</th>
<th>Role</th>
</tr>
</thead>
<tbody>
<tr>
<td>Child</td>
<td>+</td>
<td>+</td>
<td>Visitor</td>
</tr>
<tr>
<td>Adult</td>
<td>+</td>
<td>+</td>
<td>Maintenance-Worker</td>
</tr>
<tr>
<td>Elderly Person</td>
<td>+</td>
<td>+</td>
<td>Administrator</td>
</tr>
</tbody>
</table>

Combinations of these parameters in a profile are listed in Table 6.1. As the preferences on logical paths are subjective to humans, only the assumed associations are presented between user profiles and the priorities of routing criteria. In this implementation, a user is allowed to specify her/his priority order.

The parameter size is used to compute accessible geometric paths for a given user. As mentioned in Chapter 5, different user sizes correspond to distinct geometric networks and paths. More details about the use of the size parameter will be introduced in Section 6.3.2.

§ 6.2 Generation of an INSM for tested data

This section shows the generation of an INSM with all the datasets presented above. Section 6.2.1 introduces the general procedure for INSM generation. Section 6.2.2 explains how to transform input data into an INSM. Based on the logical networks of a transformed INSM, Section 6.2.3 distinguishes between complex and simple buildings by using the ratio of Connector nodes to all the nodes of a logical network.

§ 6.2.1 General procedure

The generated data are stored in a Bentley DGN file (a MicroStation format) [Sys16c] that contains floor plans of a building with different semantics stored in MicroStation layers. For instance, a ‘Space’ layer contains the geometry of indoor horizontal spaces (i.e., HorizontalUnit (HU)), a ‘VU’ layer for vertical passages (i.e., VUs), and an ‘Opening’ layer stores all Openings (OPN) of the building. Similarly, the ‘Obstacle’ and ‘POI’ layers are designed for the classes Obstacle (OBS) and PointOfInterest in the INSM. Specifically, different subtypes of VU are clarified, i.e., Stair, Escalator, and Elevator.

This subsection elaborates on the transformation of raster image and vector-based floor plans, IFC and CityGML LoD4 models to an INSM. Theoretically INSM can represent either 3D geometry (volumes) or 2D geometry (polygons in the 3D space) for indoor spaces. For tests in this research, polygons are adopted to represent spaces.
The transformation is based on the manual digitalization of non-digital materials (i.e., paper/image) and semi-automatic geometric computations of digital materials. The general procedure of INSM generation includes ten steps, which are listed below:

1. Load an original dataset into MicroStation V8i.

2. Generate 2D/3D floor surfaces of HorizontalUnit (HUs) and OPNs as polygons. Create a new larger OBS when two or more OBS polygons overlap.

3. Add space semantics to the polygons, and store them in different MicroStation layers (e.g., ‘Space’ and ‘Door’).

4. Create 3D polygons to represent vertical passages between different floors, assign the VU semantics to the polygons and store them in a MicroStation layer.

5. Add a VerticalConnector (VC) for each VU. As the VC is not available in the original building models, the geometry of the VC is manually created for each VU.

6. For two connected spaces with a void doorway, create polygon geometry and the Door semantics to this doorway. Store the polygon in the Door layer.

7. Ensure Doors and Windows overlap the corresponding spaces. Though there is no overlap between spaces of the INSM, this step is designed to support automatic detection of the connection of spaces.

8. Create OBSs as polygons and ensure every OBS is contained in a related space.

9. Generate POIs as points and store them in the MicroStation layer POI. POI can be manually selected, or be generated by input coordinates.

10. Assign the semantics of HC and END to HU spaces, according to their definitions in the INSM.

As building data are not acquired with accurate coordinates in the absolute systems (e.g., the geographic coordinate systems), the created INSM only refers to local systems. Using the local coordinate systems would not impair the computation and visualization of the two-level routing results. The length of geometric paths can be compared in local coordinate systems. Alignment to outdoor coordinate systems is left to future work (see Chapter 1).
FIGURE 6.2 Vermeerstoren. (a) Image; (b) Digitalized Doors (in green), Windows (in blue) and Walls (in black); (c) Navigable spaces such as HUs (in red); (d) VUs including Stairs (slope) and an Elevator (vertical polygon); (e) VCs neighbouring the VUs; (f) Doors and Windows are extended to overlap NUs; (g) OBSs (the purple polygon); and (h) POIs close to the OBSs (black dots).
§ 6.2.2 Transformation of the INSM from tested data

In this subsection three alternative data sources are used in the above proposed procedure to create content for the INSM: 1) floor plans; 2) IFC and 3) CityGML. Currently this procedure is semi-automated. At the end of this subsection, cases of full automation will be discussed.

Floor Plans

As mentioned previously, indoor spaces are manually created according to the INSM from floor plans. Given the Vermeertoren as an example, Figure 6.2 presents the whole mapping procedure of floor plans. In step 1 the image (JPG format) of the Vermeertoren is loaded (Figure 6.2a) in MicroStation V8i.

In Step 2, Doors (in green), Windows (in blue) and Walls (in black) are digitalized (Figure 6.2b). Based on the wall boundaries, HUs (Figure 6.2c) are created. In the implementation for this research, Walls are only used to locate the HU boundaries.

In Step 3, HUs, Doors and Windows are stored in MicroStation layers ‘Space’, ‘Door’ and ‘Window’, respectively (Figure 6.2b and 6.2c).

In Step 4, VUs are manually created in floor plans. Although the boundary of indoor spaces can be semi-automatically traced from architecture plans along with some rules [Wu15], VUs are difficult to be traced. According to the entrances of a staircase, it is symbolically created (i.e., no actual steps) as a 3D polygon (Figure 6.2d) and stored in the ‘VU’ layer with the Stair attribute. Similarly, an elevator is created between two floors (Figure 6.2d), and it is put in the ‘VU’ layer with the Elevator attribute.

In Step 5, polygons with the VC semantics are manually created to link VUs (Figure 6.2e). The VCs are stored in the ‘Space’ layer with the VC attribute.

In Step 6, doorways are added between the VC and Stair, because they connect to each other but they have no physical doors. The doorways are put in the ‘Door’ layer with the ‘Doorway’ attribute.

In Step 7, OPNs (e.g., Doors) are extended to overlap corresponding HUs (see Figure 6.2f). In this way, the connection between HUs and OPNs can be automatically detected, and the connectivity between HUs can be computed to generate a logical network. Note that this step is a temporary measure for automation purpose, and the subdivision of the building still results in indoor spaces without overlaps or gaps. In this test, the accurate building subdivision is kept.

In Step 8 two pillars are created as OBSs contained in two different HUs (Figure 6.2g). The OBSs are used in geometric routing.

In Step 9 a POI is specified close to the pillar (see Figure 6.2h), because the pillar is a salient object for users. In the MicroStation layer of ‘POI’, a user can add her/his preferred locations as POIs or provide coordinates to obtain this POI automatically.

In Step 10, HC s and Ends are distinguished automatically. From Step 1 to 9, the semantics of HU, OPN, VU, OBS and POI are obtained. However, for all the HUs in the
‘Space’ layer, their subtypes of HC and End are not differentiated. Based on their definitions, a simple procedure (see Figure 6.3) is used to assign the semantics of End and HC to polygons in the ‘Space’ layer.

**IFC**

The mapping of the IFC model to the INSM follows the general procedure (as that used for floor plans). Indoor spaces in the IFC model are separated solids. Instances of the IfcSpace correspond to NavigableUnit (NU, including HUs and VUs), and instances of IfcDoor are related to Doors of the INSM (see Figure 6.4). The geometry of vertical passages (e.g., IfcStair) and IfcDoor is a composite of solids. As it is difficult to trace the accurate steps of IfcStair instances automatically, instances of IfcStair are converted to the ones of the Stair in the INSM (Figure 6.4). Figure 6.5 presents the generation of geometry of an IfcStair instance: the stair boundary is traced with a 3D polygon without steps, which would not influence routing directions. Figure 6.6 presents the INSM of Residence.

Compared to floor plans, the IFC model has different implementation details in Step 2. In Step 2, HUs, OPNs and OBSs in each floor can be automatically derived (see Figure 6.4). This automation requires only several IFC classes such as IfcSpace, IfcDoor, and IfcFurnishingElement. The IfcSpace and IfcDoor correspond to the classes of NU and Door in the INSM, respectively. The IfcFurnishingElement is related to OBS in the INSM. The geometry of IfcSpace is a simple solid, but IfcDoors and IfcFurniture contain a collection of different solids. In MicroStation the following surfaces are extracted: the bottom surfaces of IfcSpace instances and the bounding boxes of IfcDoor and IfcFurniture instances. These extracted surfaces are put in layers of NUs, Doors, and OBSs, respectively. When surfaces of OBSs (e.g., a desk and chairs around it) overlap, they are merged into a larger one (e.g., the far right red polygon in Figure 6.6).

In Step 6, besides doorways between the VC and Stair, doorways are also added between the connected HUs. In the original dataset, two IfcSpace instances are connected without an IfcDoor instance in between them (Figure 6.7a). This problem results from the subdivision of Residence: the two spaces represent the whole hall but they are separated in the dataset. Thus a doorway is generated between the two HUs after the INSM of Residence is created (Figure 6.7b).

**CityGML**

The CityGML LoD4 model is mapped to the INSM following the general procedure presented previously. Figure 6.8 illustrates the mapping between CityGML LoD4 and the INSM. Surfaces for HU, OPN and OBS, Stair and Elevator are manually created (Figure 6.9). According to the latest version v2.0.0 of CityGML, instances of Room can be
FIGURE 6.4  The associations between the IFC classes and INSM classes, i.e., the mapping of the two models.

FIGURE 6.5  Creation of INSM geometry from an IfcStair instance. (a) The stair as a composite solid; and (b) The stair as a 3D polygon.
FIGURE 6.6 The INSM of Residence. Grey polygons represent NUs, green ones are OPNs, and red ones are OBSs.

FIGURE 6.7 An example of a NO explicit door. (a) No IfcDoor instance between two IfcSpace instances; and (b) A Doorway is added.

mapped as either HU or VU based on the specified attribute values. In addition, Int-BuildingInstallation instances can be either an interior stair or an OBS (e.g., a fixed installation). Such ‘Multiple-to-One’ relationships usually complicate or impede an automation procedure to obtain surfaces of HU, VU and OBS. Thus the above spaces are manually created by copying the available geometry in the CityGML files.

Towards full automation of INSM generation

Although the automatic mapping is outside the scope of this research, some ideas for facilitating the automation are discussed. First, closed spaces with semantics should be automatically derived from architecture plans by applying some rules. In research [Wu15] related to this thesis, an initial automation is developed to trace closed spaces bounded by openings and their adjacent walls. In this method windows and doors of input floor plans are drawn, and the unnecessary details of the plans are cleaned manually. This method can recover indoor spaces for regular structured buildings, but re-drawing rules and multiple thresholds for the automation need to be specified manually based on specific scenarios. Another building reconstruction method designed by Dominguez-Martin et al. (2015) presents the automation of a building reconstruction with a part of geometric details [DMvOFH+ 15]. They take a CAD file in three views (top, front and side) as input, generate contours of the three views, and then extrude a volume (i.e., space) from these contours. But this method still cannot reconstruct a complete geometric representation of building.

Second, intelligent use of semantics can help in the automatic mapping. As mentioned above, the mapping from CityGML to the INSM includes many ‘multiple to one’ cases
FIGURE 6.8  The conceptual mapping between CityGML and INSM.

FIGURE 6.9  The INSM of the old OTB building. The stairs and the elevator are manually digitalized as 3D polygons.
(see Figure 6.8). For example, a Room instance of CityGML can be a HU or an Elevator in the INSM. When tags of Room instances are clearly attached by automatic classification based on certain rules (e.g., shapes or other geometric characteristics), an automation method can be developed for this mapping.

Third, the reconstruction of vertical passages in the three types of adopted data can be automated by approximating the geometry. For instance, it may be difficult to extract an accurate 3D surface automatically for vertical passages if the geometry of the vertical passage is represented by a composite of multiple surfaces (i.e., steps). An automation can be developed to fit a stair with several slopes, which omits steps of the stair as they are not important for routing. The geometry of an elevator in IFC or CityGML is represented by solid or multiple surfaces. In such a case, one of the wall surfaces of the elevator can be extracted automatically to represent the elevator.
For the coming tests, *complex* and *simple* buildings need to be distinguished, which reflects the complexity of their related INSMs. To define what is a relative complex or simple building, the distribution of node degrees of logical networks is computed and compared. As mentioned in Chapter 4, the degree of a node reflects the number of other connected nodes, and the logical network is a direct graph where one node has bi-directional degrees (in-degree and out-degree). A node with the total degree value 4 connects to two other nodes in the logical network, i.e., Connector. The ratio of Connector nodes (degree ≥ 4) to all the nodes (Connector ratio for short) is calculated for a logical network. A high ratio indicates that many Connectors exist and may increase path choices between two locations in the building. In this sense, buildings with a high Connector ratio are regarded as complex.

**TABLE 6.2** Connector ratio of logical networks for all the INSMs. Schiphol Airport has the highest ratio (35.79%) and Residence includes the lowest ratio (10.00%).

<table>
<thead>
<tr>
<th>Building</th>
<th>Nodes more than 4 degrees</th>
<th>All nodes</th>
<th>Connector ratio (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Schiphol Airport</td>
<td>34</td>
<td>95</td>
<td>35.79</td>
</tr>
<tr>
<td>OTB</td>
<td>26</td>
<td>139</td>
<td>18.71</td>
</tr>
<tr>
<td>MFA</td>
<td>45</td>
<td>131</td>
<td>34.35</td>
</tr>
<tr>
<td>Residence</td>
<td>2</td>
<td>20</td>
<td>10.00</td>
</tr>
<tr>
<td>Vermeertoren</td>
<td>11</td>
<td>45</td>
<td>24.44</td>
</tr>
<tr>
<td>CNICU</td>
<td>3</td>
<td>13</td>
<td>23.08</td>
</tr>
<tr>
<td>BK</td>
<td>14</td>
<td>91</td>
<td>15.38</td>
</tr>
</tbody>
</table>

The Connector ratio is computed for logical networks of all the INSMs. Schiphol Airport, OTB, MFA, Residence, Vermeertoren, CNICU and BK contain ratios of 35.79%, 18.71%, 34.35%, 10.00%, 24.44%, 23.08% and 15.38%, respectively (see Table 6.2).

Then complex and simple buildings are defined based on the above results. A building is complex when its Connector ratio is higher than 30%; otherwise, the building is a simple one. As Schiphol Airport has the highest ratio (35.79%), it is regarded as the most complex of all the others. MFA is also considered complex with the second highest ratio (34.35%). The other buildings with low ratios (Residence, OTB, Vermeertoren, CNICU and BK) are regarded as simple buildings.
§ 6.3 R**outing**

§ 6.3.1 Logical network

This section discusses the implementation of routing on a logical network. Logical networks are created for two complex and three simple buildings: MFA (complex), Schiphol Airport (complex), Residence (simple), OTB (simple), and Vermeertoren (simple).

*FIGURE 6.10* The logical network of Residence. (a) The INSM of Residence; and (b) The derived logical network.

Based on the *INSM* of Residence (Figure 6.10a), a logical network is derived automatically from the connectivity of spaces. NUs are connected when they all link to a Door. In addition, multiple Doors between two NUs are represented by only one logical edge. According to the definition of logical networks (see Chapter 3), the logical network is represented by a directional connectivity graph (see Figure 6.10b). The *INSM* semantics of nodes are differentiated in distinct colours.
FIGURE 6.11  The logical network of MFA. (a) The INSM shown in an aerial view and in a rotated view. 3D polygons in the vertical direction are VUs, and (b) The logical network.
Similarly, the logical network of MFA is obtained (see Figure 6.11a). In this logical network, some VU nodes seem like End nodes (see Figure 6.11b) because the other related floors are not presented. As mentioned before, MFA is a complex building, thus multiple logical paths may exist between two spaces.

The logical networks of OTB and Schiphol Airport are given in Figure 6.12 and Figure 6.13. The one of Schiphol Airport is used to test the Floor strategy. The other two criteria applicable to single floors are tested in MFA: the minimum NU and central HC are applied in a priority order. On both logical networks, the routing option L1.1 (see Chapter 3) is implemented. Details of the tested routing criteria are below:

- **Floor strategy.** The Floor strategy helps a user to arrive first at the floor of a destination. This means VUs near the user are located first.

- **Minimum NU.** The Minimum NU criterion aims at a logical path through the fewest number of spaces.
• **Central HC.** The Central HC criterion collects the high accumulated centrality of HCs in a logical path without detours to the target, which ensures the horizontal motion is on the central sections (*e.g.*, main corridors) of each floor.

The Floor strategy is realized because Schiphol Airport (see Figure 6.13) is a multi-floor complex building where multiple paths can exist between two spaces on different floors. Suppose that a traveller needs to find her/his path from the entrance hall to the correct check-in point. The traveller has the following profile: \([\text{Adult, With-Wheeled-Devices, Visitor, 0.6m}]\). This is a typical profile for travellers since most of them bring some luggage on trolleys to their destinations. In this example, a size of a traveller with luggage is set to 0.6 meter(m) which is only used for geometric routing. Escalator is specified in the Floor strategy for the traveller, because it saves the traveller effort to bring the luggage to the target location. There are two resulting logical paths through Escalators (Figure 6.14).

Another routing test is presented on MFA (Figure 6.15). This is a case that routing is conducted with multiple criteria on the logical network. In the logical network of MFA (Figure 6.11), a visitor is assumed to have the following profile: \([\text{Adult, Walking, Visitor, 0.5m}]\). A priority order is set for two criteria for routing: 1) *minimum NU*, and 2) *central HC*. This order means firstly the visitor needs to walk to the target room by crossing the fewest number of spaces (*minimum NU*), and secondly she/he aims to pass through the central section of MFA (*central HC*). In this example the visitor does not change floors. After applying the criterion *Minimum NU* to the logical network, two logical paths are computed (Figure 6.15a). A unique path (see Figure 6.15b) is obtained after applying the criterion *Central HC* to the previous two logical paths. This unique path has higher node centrality and it is relatively easy to change the path at each node.

As can be observed from the images (Figure 6.14 and Figure 6.15), the above logical paths are visualized as a sequence of spaces, which helps a user to identify the path that she/he needs. In both the complex buildings of Schiphol Airport and MFA, multiple logical paths exist between the given start and target spaces. In such a case, it is necessary to select the proper logical path according to a user profile when the user requests a definite path. Using the priority order of multiple criteria is an efficient method to select logical paths.
FIGURE 6.13 The logical network of Schiphol Airport. (a) The INSM, and (b) The logical network.
FIGURE 6.14 Paths complying with the *Floor* strategy. Two logical paths are derived. The black arrows show the directions of the logical paths.

FIGURE 6.15 Routing with two criteria on the logical network of MFA. (a) Two *minimum NU* paths; and (b) The *minimum NU + central HC* path.
§ 6.3.2 Geometric network

This section presents routing on geometric networks in a single space and a sequence of spaces. Such a routing depends on the size of users. Chapter 5 mentioned that the complete geometric network of a building can be formed with geometric networks in each single space, without details of the procedure. This section presents the implementation of geometric network generation, which is influenced by obstacles and user size. As mentioned before, INSMs contain obstacle information and the designed user profile includes the parameter of user size.

The following tests are conducted:

1. In a space, routing with different sizes between the same start and target doors, to check geometric paths from the different results of obstacle grouping. For example, a visitor and a maintenance worker go to the same target location. Because the maintenance worker brings an indoor investigation device, her/his size is larger than the visitor. The routing regarding each size is related to routing option G2.1 (see Chapter 3) that needs no POIs but a user size.

2. In a space, routing when the same user moves in a space changing different sizes. For example, a person needs to pick up a luggage trolley at a specific location in a space, and then pushes it to the next location. This test is the implementation of routing option G2.3 (see Chapter 3) where POIs with different user sizes are specified.

3. Given a sequence of spaces, to form geometric networks in each space into a larger geometric network including doors and POIs. For example, a person assigns a logical path indicated by a number of spaces, and then she/he requests an accessible geometric path passing through some POIs. Thus a geometric network is created first regarding these spaces. This test is related to routing option G2.2 (see Chapter 3) that provides multiple POIs and their ordering.

The first test is to demonstrate the influence of obstacles and user size on geometric paths. In the second test, routing with two different sizes of one user is applied in one space. The third test demonstrates how a geometric path can be computed for more than one space. As mentioned in Chapter 3, all the locations represented by geometric nodes are doors and POIs. The third test also includes some windows, and they are specified as POIs since normally they are not used for transfer among spaces.

Geometric paths are computed based on two INSMs of CNICU (see Figure 6.16a) and BK (see Figure 6.16b). The CNICU contains many indoor obstacles, but the BK contains few indoor obstacles.

In the INSM of CNICU, the test is conducted for users with respect to three sizes: 0.5 m, 0.6 m and 0.8 m. The first size represents a common width of a person (Figure 6.17a). The second size 0.6m can be considered for a person who is pushing a wheelchair (Figure 6.17b), e.g., a care worker. The third size 0.8 m is used for a maintenance worker (Figure 6.17c), for example, one who needs to bring equipment to repair electrical wiring in the ceiling.

The test clearly shows the effect of user sizes on the computed paths. In the test it is assumed that users keep a distance from the walls. Therefore, a buffer of 0.5/0.6/0.8m
FIGURE 6.16 The INSM of two buildings. (a) CNICU; and (b) BK.

is created around the walls (the dashed polygons in Figure 6.18). In Figure 6.18a, the shortest path for 0.5 m lies in the middle of the Final Convex Hull (FCH, the black polygon) between two doors. Figure 6.18b presents the path for 0.6 m where a part of the path is on the edge of FCH. There is no path for 0.8 m users (see Figure 6.18c).

A user can be informed in advance about the accessible geometric path for a given size, or no available paths. As visible in Figure 6.18, there is a small difference (0.3 m) between the user sizes in figures 6.18a and 6.18c, but there is no path in Figure 6.18c.

In the INSM of BK, routing for a person with changing sizes in selected spaces is presented. Given a student with the size 0.5 m, she/he walks into the designing hall to get a trolley to move a number of architectural models, and then crosses the space to arrive at the next one (Figure 6.19). The start location is an entrance door on the left of the hall (Figure 6.20a), and the target location is the door on the right of this hall (Figure 6.20b). The size of the student increases to 1.2m after she/he arrives at the trolley place. After the student walks into the next space, the user size does not change. To realize this case, a POI is added at the location of the trolley (the black point in Figure 6.20a). A path is computed first for the original size 0.5m by grouping related obstacles. Then another path is computed for the changed size 1.2m from the POI to the target location. The two geometric paths are combined as one for both the sizes before and after the change (Figure 6.20c). A geometric network (Figure 6.20d) in the next space for the size 1.2m is also generated.

As described in Chapter 5, the set of all the shortest paths for a given size compose the geometric network. Normally one geometric network in a space is computed for a given user size. This network can be either created ‘on the fly’ or pre-computed. However, in the case in Figure 6.20c, it is not wise to pre-compute several geometric networks
FIGURE 6.17 Different users. (a) Common pedestrian (from www.prefast.com); (b) Care worker (from www.friendsofsavernake.org); and (c) Maintenance worker (from www.americantrainingresources.com/ptv-127.aspx).

FIGURE 6.18 Routing for the sizes of 0.5 m, 0.6 m and 0.8 m. (a) The shortest path for 0.5 m; (b) The shortest path for 0.6 m; and (c) No path for 0.8 m. The dashed polygons represent the buffer to the walls.

in the space and store them. Thus paths are computed on the geometric network derived ‘on the fly’, based on the start and target doors and the POI where the user size changes.

The following test will demonstrate how to link geometric networks in single spaces and aggregate them into a larger geometric network. Four spaces are selected to construct geometric networks sequentially. All the creation of geometric networks in these spaces is for a user size of 0.5 m. Firstly the shortest paths for 0.5m among the doors and POIs are computed in each space, and then the geometric network consists of all these shortest paths. Door-to-door paths are used for transfer among spaces since the inside of spaces is not interesting to a user. But a door-to-POI or a POI-to-door path is contained inside a space and refers to a location to be visited, such as the reception desk, the platform, or the window in Figure 6.21b. When the geometric networks in the four spaces are combined into one (Figure 6.21b), a geometric path through three ordered POIs is computed (Figure 6.21c).
FIGURE 6.19  The designing hall in BK (from tudelft-architecture.nl). It is a space with a group of obstacles where routing is conducted for a user with changing sizes.

FIGURE 6.20  Change of user size in one space. (a) Path for the 0.5 m size from the start door to a POI. The highlighted polygons represent the grouping results of obstacles; (b) Path for the 1.2 m size from the POI to the target door; (c) The final path for the user between the two doors; (d) A geometric network for 1.2m in the next space.
FIGURE 6.21 Geometric network for four spaces in BK for the size 0.5m. (a) A logical path (highlighted spaces). Space 1 is the start, and space 4 is the target; (b) Complete geometric network for the four spaces, regarding doors and POIs; and (c) A geometric path from the POI in space 1 to the POI in space 4.
FIGURE 6.22 The complete geometric network of BK for the size 0.5m. This network considers only door-to-door paths and it contains 2216 geometric edges.
Similarly, if a complete geometric network of the whole floor is needed for a user with a given size (e.g., 0.5 m), the complete geometric network can be constructed by putting together all the geometric networks in all these spaces (Figure 6.22).

§ 6.4 Analysis of the tests

This section elaborates on the results presented in sections 6.2 and 6.3, discusses the used classes of INSM and suggests future implementation of one-level routing.

As mentioned in Section 6.2.1, the main classes of INSM (HU, VU, Opening, Obstacle and PointOfInterest) are implemented as MicroStation V8i layers. To generate the logical network, NU and OPN are mostly used. In particular, VC, HC, and End, the subclasses of HU, are automatically determined by a procedure (see Figure 6.3). Other bottom-level subclasses of the above classes, such as Stair, Elevator, Escalator, Door, FacadeWindow and InteriorWindow, are attached to instances in related layers as attributes in MicroStation V8i. The aggregation classes (HS, VS, NBS, BP and BLD) are omitted for the generation of logical networks.

Though not all INSM classes are necessarily used for the generation of logical networks, the aggregation classes should be adopted when a hierarchical structure of a building needs to be created for other applications. These classes are designed to increase the abundance of semantics of INSM. For instance, when a logical network for floors is required, HorizontalSpace (HS) instances representing floors and VerticalSpace (VS) instances linking the floors would be used to generate this logical network (see Figure 3.5 in Chapter 3). According to a functional subdivision, some conceptual regions (without physical boundaries) of a building can be depicted as well. In such cases, HS and/or VS can also represent the collection of these regions. Thus HS and VS can be applied to distinguish between different granularities. NavigableBuildingSpace, the superclass of HS and VS, is useful for inquiries about the free space in a whole building. The class BuildingPart can depict some special cases of buildings, such as several building parts connected by sky bridges. In general, these aggregation classes can be applied to different granularities for a hierarchical model of the building. But in this thesis, the focus is only on two granularities, i.e., independent spaces physically bounded by walls and the details in the spaces.

For the generation of geometric networks, three main INSM classes – NU, OPN and OBS (one of the subclasses of NonNU) are adopted. Wall, another subclass of NonNU, would not influence the created geometric networks since its instances are not accessible. Although they are digitalized (see Figure 6.2b), they are used only to support the creation of HUs. As a Wall is separate from navigable spaces, the two-level routing would not be impacted by the absence of it. In the INSM model Window and Door are subclasses of OPN. Door is mostly used in the previous tests, while Windows are also applicable as POI. Whenever a user plans to consider windows for indoor routing (e.g., in emergency cases), they could be added in a geometric network. Figure 6.21 presents an example of a window as a POI. In general, both Door and Window can be specified as POIs.

For the creation of geometric networks, the aggregation classes (e.g., VerticalSpace(VS), HorizontalSpace(HS), NavigableBuildingSpace(NBS) and BuildingPart(BP), see figures
3.5 & 3.6) in the conceptual data model of INSM are also not used. In a word, all these classes enhance the representational ability of INSM with cases of composition spaces, but in the tests they are not used for the generation of logical and geometric networks. Only a part of INSM is used to conduct the routing. These aggregation classes can be useful for some other applications (e.g., needing a detailed hierarchy of spaces).

In a building, different users can reach distinct parts of the building in the form of INSM. Even for public buildings, it is not recommended to expose all the building information to all. In this case, INSM can be used to decide which parts of a building are to be exposed and to what type of user. For instance, the image of Schiphol Airport contains only a part of the airport information for the public. With regard to data safety, a user can access building information in light of her/his permission level. A part of a building model needs to be selected for a specific user according to the user’s profile. For example, a visitor cannot access electricity wells for maintenance. Thus all the spaces for technicians are not open to the visitor. Paths for the visitor are computed on the basis of a subset of the overall INSM. In contrast, an administrator gains the complete view on the INSM since she/he has the highest access permission in the building.

The test results of logical networks (see Section 6.3.1) indicate there can be multiple logical paths between two spaces, even after applying a priority order of criteria. At present, all the related logical paths for a user are computed. To handle multiple paths, three methods can be applied: 1) to provide all the logical paths to a user; 2) to ask a user to select one of them; 3) to compute first all their corresponding geometric paths, and then select the logical path regarding the ‘best’ geometric path (e.g., it can be the ‘shortest’ one among them).

The main advantage of using logical networks is that spaces are represented conceptually. For instance, a coffee corner is an open space without a physical boundary, but it can be considered an SOI and be represented in a logical network. Humans can understand the logical path as verbal or textual descriptions, such as ‘the current space is the entrance hall, the next space is the long corridor, and then the left stair...’.

Logical networks are derived from INSM according to connectivity relationships among spaces. Edges of logical networks represent the connectivity of spaces regardless of one or multiple doors. For instance, two spaces are connected via three doors (Figure 6.23a) and in the logical network the three doors are represented by two directed edges (see Figure 6.23b). Although in a logical path a user can understand which spaces would be visited, the door to pass is not specified. This can be done in the two-level routing approach, where a user is given the exact door to use.

To consider multiple doors connecting two spaces in a logical path, multiple edges should be allowed to connect the two spaces (Figure 6.23). Such a network is called multigraph [Die10]. However, such networks still do not help to mediate a specific door. It simply provides the information that more options are possible. Therefore, the logical network used in this research does not allow multiple edges between two spaces.

The tests in Section 6.3.2 show that a geometric path between two doors of a space is influenced by the distance among obstacles and those between obstacles and physical boundaries of space. For a user whose size changes while moving in a space, the creation of a complete geometric network in advance is not recommended, because the
network heavily depends on the motion (the start location, intermediate POIs and the target location) and the user sizes. The test result (Figure 6.21c) suggests that a good option would be to compute a geometric path only ‘on the fly’ in the space. In this case it is unnecessary to maintain two different networks for the two user sizes in the same space.

In this implementation, routing on geometric networks is conducted without consideration of user profiles (except user size). As given in Chapter 5, geometric paths are computed with the Dijkstra algorithm, i.e., the shortest path in a geometric network. Doors in between multiple spaces are determined by the shortest path. However, geometric paths can also be computed in each space and be formed into a whole path, when the routing takes into account a user’s needs on doors and turns. Figure 6.24 is an example of a user that is given the option to select a geometric path in a space. The red space is the start space and the blue one is the target space. There are three geo-
metric paths inside the current space. The user selects the middle path (Figure 6.24) since it leads to the closest door. In this way, given a logical path, paths in each space can be determined by a certain criterion (e.g., the closest door) and then be aggregated into a complete geometric path.

§ 6.5 Summary

Section 6.1 presents the used software tools, data and user profile format for implementing one-level routing. Section 6.2 presents the creation of INSM based on the adopted real building data. A general procedure of INSM generation is elaborated and illustrated with floor plans, IFC and CityGML LoD4 models. This procedure requires manual effort. Section 6.3 presents the results of logical and geometric routing with designed scenarios of real buildings. Section 6.4 discusses the use of INSM based on these routing results and elaborates on further cases about both logical and geometric routing.

This chapter partially answers the following research sub-question:

6. How are the new proposed user-related paths implemented and applied to realistic cases?

Section 6.3.1 presents the implementation of three proposed routing criteria as typical examples (Floor strategy, Minimum NU and Central HC). The user-related information is reflected in the designed user profile (see Section 6.1.2) including parameters of age, mobility, role, and size. Combinations of these parameters for a user can be linked to specific path preferences in the logical network. In the implementation an association is assumed between a user profile and her/his preference of logical paths.

Section 6.3.2 illustrates the implementation of geometric routing in three realistic scenarios: 1) separate users with distinct sizes; 2) one user with changing sizes; and 3) formation of a geometric network from a sequence of spaces. Tests show that the size of a user critically influences the shape of a geometric path. In the proposed routing method (see Chapter 5), both the cases of distinct user sizes and one user with changing sizes are tested. According to the interaction of a user, an aggregated geometric network can be formed from any selected spaces, which shows the flexibility of generating user-related geometric networks. This flexibility stimulates a user to give more consideration to the abstract level (about spaces), and obtain customized logical and geometric paths.

For the same research sub-question, the next chapter will introduce the implementation of the two-level routing, which integrates both the logical and geometric routing for different users.
Realization of two-level routing

In this chapter, three factors are considered for the implementation of two-level routing: 1) the way to compute the data content for three models (INSM, and two derived models – the logical network and the geometric network); 2) the place to store the three models (i.e., data organization); and 3) user interaction. In Section 7.1 five cases are presented with client-server architecture that can be implemented for desktop and mobile platforms: one of them is for desktop applications and the others for mobile applications. Two of the implemented cases are elaborated, i.e., the desktop application (Section 7.2) and a mobile application (Section 7.3). To cater for different user sizes, the desktop application derives geometric networks on the fly, and two-level routing is conducted with three typical routing options (C3.1, C3.3 and C3.6) that were introduced in Chapter 3. Option C3.1 is routing regarding a constant user size without SpaceOfInterest (SOI)/PointOfInterest (POI); Option C3.3 refers to changing sizes with ordered POIs; Option C3.6 is about a constant user size with ordered SOIs/POIs. Routing results for two different buildings are compared and analysed in performance and the involved number of points used for geometric networks.

A two-level routing mock-up is implemented with Bentley Systems Mobile SDK in a Bentley mobile application Navigator Mobile. With a lightweight dataset of logical and geometric networks, the two-level routing function has been developed in the simulator of the Navigator Mobile. The exchangeable data format i-model [Sys16c] of Bentley Systems is used for this mock-up. The conceptual models of the logical and geometric network (see Chapter 3) are implemented as XML-based data schemas for the mock-up. The logical and geometric networks of the building are derived first in the desktop application, and they are stored in ECSchema files (a XML-based file of Bentley Systems) [Sol16]. Finally, the INSM, the ECSchema files and the 3D building model are wrapped into an i-model dataset. Section 7.3 will present routing on the logical and geometric networks in the mobile application. The two-level routing mock-up demonstrates the feasibility of the two-level routing on a mobile device. This mobile mock-up stores one complete geometric network for users with a given size. The testing results show the two-level routing can be independently conducted in this mobile mock-up. Section 7.4 discusses the test results and provides the future improvement on the two-level routing approach. This chapter is closed with a summary in Section 7.5. This chapter is based on three author’s own publications: [LZ11a, LZ13b, LZ13a].

§ 7.1 Factors considered for realization

Model creation. For the implementation, the first factor to be considered is how the data content for INSM and two sub models (the logical network and the geometric network) can be created. The INSM is too complex to be obtained automatically, thus it is worthwhile being created once and stored. The logical network of all the spaces in a building can be created once and maintained, because in normal conditions these spaces do not change. For certain cases, a part of these spaces needs to be selected to form a specific logical network for a user (e.g., a visitor with a wheelchair). Access permission and accessibility for different users are stored as an attribute of the edges. In
this case, a sub-logical network can be selected ‘on the fly’. Therefore, a logical network for users can be either pre-computed or derived ‘on the fly’.

Similarly, for a given user the complete geometric network can be maintained for a building. According to the building owners, they may only need several geometric networks for different user sizes in common situations. They also need to adopt new user profiles to derive suitable geometric networks ‘on the fly’. In addition, when a logical path is re-computed in the light of a changed user size, only the related spaces contribute to the creation of the corresponding geometric network. In this case, the geometric network is computed ‘on the fly’ as well. Thus the geometric network can be either pre-computed (for one or more sizes) or derived ‘on the fly’ (for different sizes).

Model storage. The second factor is where the three models can be stored. The Client-Server architecture provides different possibilities to deploy the three models for two-level routing. Specifically, INSM and the logical and the geometric networks can be flexibly computed and stored in different hosts. The generated data are stored in DGN (desktop) files and i-model (mobile) files of Bentley’s format functionally similar to a Database Management System (DBMS) [Sys16c].

For example, one can store the INSM in a user’s mobile device for visualization, develop an application in a server to compute paths on the logical and geometric networks for the user, and send the paths back to the mobile device. For a building, a complete geometric network is always larger than a logical network, and the INSM of a building can be larger than the geometric network. An alternative is to store all these models in a server, and the mobile device can then download and visualize the necessary parts of the INSM and computed paths.

Generally, a thin client is considered only for the visualization of paths in the building, without keeping all three models (INSM, and the logical & the geometric networks) and path computation. The data for visualization is downloaded from the server side. A thick client is dependent on its functionality: it should be able to locally compute the two-level routing results customized for a user’s profile. In order to process the routing tasks, a thick client requires higher processing ability than a thin client. The routing computation on the thick client includes the creation of logical and/or geometric networks and path computation on these networks. For a mobile application, in addition to routing computation the data amount should also be considered for a client, because considerable storage may increase the load for mobile devices (e.g., the INSM of a group of buildings in a campus). Along with the increasing computational ability of mobile devices, the thick client would be more affordable and users can benefit from routing on mobile devices in the offline phase.

Five cases are presented for implementation of two-level routing with the Client-Server structure (see Table 7.1). The first two cases have been implemented in this thesis. They are both thick clients, i.e., both the logical and geometric routing is conducted on the client. The main difference of the two cases is if they are desktop (Case 1) or mobile (Case 2) clients for implementation (see Table 7.1). The other three cases are thin clients, which means the client only stores lightweight data (INSM and/or a logical network) or no data, and receives computed geometric paths from the server. In the third case, the server owns the complete geometric network while the client maintains only the logical network and the INSM. The fourth case adds the INSM, and the logical and the geometric networks on the server, and thus the client does not conduct any compu-
### Table 7.1: Five cases of the implementation of two-level routing in a Client-Server structure. '+' is stored and '-' is not stored or derived on demands.

<table>
<thead>
<tr>
<th>Case</th>
<th>INSM Data</th>
<th>Logical Network</th>
<th>Geometric Network</th>
<th>Type</th>
<th>Application</th>
<th>Implemented in this thesis?</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>+</td>
<td>+</td>
<td>-</td>
<td>Thick Client Desktop</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>Thick Client Mobile</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>+</td>
<td>+</td>
<td>-</td>
<td>Thin Client Mobile</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>Thin Client Mobile</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>-</td>
<td>+</td>
<td>+</td>
<td>Thin Client Mobile</td>
<td>No</td>
<td></td>
</tr>
</tbody>
</table>

In Case 1, the INSM and the logical network are stored in the client, and the geometric network is computed 'on the fly'. This case is suitable for a desktop application since the desktop is capable of deriving geometric networks in real-time for the two-level routing. For example, routing services are requested by an airport administrator who needs to check suitable paths for different people. Logical paths and geometric paths are all computed in the desktop client. It provides indoor routing and only serves a local user. Then Case 1 can present the requested path on the desktop computer.

Case 2 is similar to Case 1 but it pre-computes and stores geometric networks for certain user sizes in the client. Case 2 is implemented as a thick client. This case can be applied to mobile devices. Suppose that the facility manager of an airport wants to find her/his way using a tablet. As all three models are stored on the mobile device, the two-level routing can be conducted in the offline phase (without the internet), which is quite convenient for specific users (e.g., an employee in a factory without a WiFi connection). Thus the application in the tablet provides indoor two-level routing for the user. Case 1 and Case 2 are implemented and will be introduced in Section 7.2 and 7.3, respectively.

In Case 3 the client only stores the INSM and the logical network, while geometric networks (for different user sizes) are pre-computed and stored on the server side (See Figure 7.1a). This case is a thin client since the two-level routing is conducted on the server, which can be a mobile application. For example, the mobile application sup-
ports visualization of the logical network, and a user is familiar with the indoor environment and she/he wants to specify logical paths on demand. The user-specified logical paths can be sent to the server to generate geometric paths, then the geometric paths are sent back to the user on the client. In the mobile client an interface should be developed for a user to input preferences on routing (e.g., to select routing criteria and their priority order, to select SOIs and POIs in the building model or input space ID and/or POI coordinates). Also, the start and target locations are specified by the user via this interface. The selected routing criteria of logical paths are sent to the server, and then the server will compute the logical path and the corresponding geometric path and return this result to the user.

In Case 4 the client stores none of the three models (Figure 7.1b). INSM, the logical network and pre-computed geometric networks are stored on the server side. As a thin client, Case 4 is suitable for a mobile application. In such a case, in the mobile client a
user needs the interface to specify the start/target by pre-defined textual references (e.g., names) and routing preferences, and to download and visualize the final paths and necessary parts of the INSM (e.g., some spaces).

In Case 5 the client does not store any data, while the server derives the geometric network in real time on demand (Figure 7.1c). This case is an example of a thin client, which also can be adopted for a mobile application. Cases 3, 4 and 5 are not implemented in this thesis.

**User interaction.** The user can interact with a routing application to adjust a path. A user needs to specify the start and target spaces and/or locations. In all the cases in Table 7.1, the user needs to input the user profile, or specify routing criteria for logical paths and the user size for geometric paths. As mentioned in Section 3.4.3, a user can also specify SOIs and POIs, and the sequence of the SOIs/POIs to be visited. Additionally, a user can supply different sizes for herself/himself changing at specific locations (POIs). After all the information is provided, a complete accessible path(s) is computed for these given sizes via these POIs.

In the application, a user can decide whether routing can be solved only with a logical network. The two-level routing is not necessarily applied to every routing request. For instance, a geometric path is not needed when the user considers a logical path is simple and she/he can follow it either because the user is familiar with the building or because she/he prefers general path descriptions. But a user can also manually select a logical path and request the related geometric path. Such a situation can be realized by a Client-Server application as in Case 3: the user selects a logical path in the mobile client and sends the logical path to the server, then a geometric path is computed by the server and sent back to the client.

---

### § 7.2 Desktop application

To address Case 1 in Table 7.1, this section introduces two-level routing implemented on a desktop computer. The **Thick client** application is implemented in a desktop computer which manages all three parts (the INSM, and the logical and the geometric networks). In this application, the server can be skipped since the computation is conducted locally.

An application is developed in the **MicroStation V8i**. It collects spaces of INSM, detects connectivity of the spaces, creates and stores logical network, and creates geometric networks on the fly. Specifically, INSM is stored as a **DGN** file which contains the semantics and geometry of a building. The derived logical network is stored in this **DGN** file as well. Geometric networks are created 'on the fly' during the routing computation. In this implementation, all the logical paths are visualized by a sequence of highlighted spaces in the INSM.

To evaluate the performance of the two-level routing approach in a complex and a simple building, three typical options for indoor navigation are selected and implemented in the old OTB building (OTB for short) and Schiphol Airport, i.e., options C3.1, C3.3 and C3.6 (see Table 7.2).
1. Option C3.1 refers to a case where the user does not provide preferences. It requires no SOI or POI, thus a user without an obvious preference would select the option to automatically obtain logical and geometric paths. This option is implemented in the two buildings to compare the number of logical paths.

2. Option C3.6 refers to the interaction of users with the routing application. This option allows the user to specify SOIs and/or POIs. This option demonstrates the forms of logical and geometric paths in the two buildings, and analyzes their differences.

3. Option C3.3 refers to cases that involve changes of a user size. Such a path is a combination of two geometric paths: 1) the first part from the start to a specified POI with size 1; and 2) the second part from the POI to the target with a new size 2. Practically the same type of computation is conducted twice for the two parts before and after the POI, but with different sizes. Option C3.3 is implemented in the two buildings and used to compare the time costs and involved nodes of the routing results.

User profiles are used in these implementations (see Chapter 6). Four parameters are maintained to reflect the user profile (i.e., age, mobility, role and size). They are linked to related criteria for routing in the logical network. Table 7.2 lists the implemented routing options, user profiles and related routing criteria in the logical network.

<table>
<thead>
<tr>
<th>Tested option</th>
<th>User profile</th>
<th>Logical path criteria</th>
</tr>
</thead>
<tbody>
<tr>
<td>C3.1</td>
<td>[Adult, Walking, Visitor, 0.5m]</td>
<td>Floor strategy + Central HC</td>
</tr>
<tr>
<td>C3.3</td>
<td>[Adult, With-Wheeled-device, Maintenance Worker, (0.8m, 0.5m)]</td>
<td>SOI strategy</td>
</tr>
<tr>
<td>C3.6</td>
<td>[Adult, Walking, Visitor, 0.5m]</td>
<td>SOI strategy</td>
</tr>
<tr>
<td>OTB</td>
<td>C3.1</td>
<td>[Adult, Effort-Saving-Motion, Visitor, 0.5m]</td>
</tr>
<tr>
<td>C3.3</td>
<td>[Adult, With-Wheeled-Devices, Administrator, (0.5m, 1.2m)]</td>
<td>SOI strategy</td>
</tr>
<tr>
<td>C3.6</td>
<td>[Adult, Walking, Visitor, 0.6m]</td>
<td>SOI strategy</td>
</tr>
<tr>
<td>Schiphol</td>
<td>C3.1</td>
<td>[Adult, Walking, Visitor, 0.5m]</td>
</tr>
<tr>
<td>C3.3</td>
<td>[Adult, With-Wheeled-device, Maintenance Worker, (0.8m, 0.5m)]</td>
<td>SOI strategy</td>
</tr>
<tr>
<td>C3.6</td>
<td>[Adult, Walking, Visitor, 0.5m]</td>
<td>SOI strategy</td>
</tr>
</tbody>
</table>

In the implemented applications, distinct colours are used for the start and target spaces, logical paths and SOIs as in Chapter 3. Specifically, the Start spaces are in light purple; the Target spaces are in light yellow; the SOI is presented in cyan pink; the logical paths are shown with green spaces; and the POIs are depicted by black dots.

§ 7.2.1 Routing without SOIs and with one size

In the OTB building, option C3.1 and C3.6 are implemented for a visitor. The user profile is [Adult, Walking, Visitor, 0.5m] (see Table 7.2). The scenario for option C3.1 is that the user at an employee’s office on the third floor intends to find the main entrance. In this case, the Floor strategy is applied to compute the logical and geometric paths for the user automatically.

The number of logical paths is compared for OTB and Schiphol Airport. The logical paths of the Floor strategy in OTB starts from a space on the third floor and lead to a space.
on the first floor. This logical path crosses the stair close to the start space (see Figure 7.2a). As it is only one path, a geometric path is computed based on this logical path (see Figure 7.2b).

![Diagram](image)

**FIGURE 7.2** Implementation of routing option C3.1 in the old OTB building. (a) The logical path of the Floor strategy; and (b) The geometric path related to the Floor strategy.

Option C3.1 on Schiphol Airport illustrates the different results. In the scenario is a traveller with luggage who needs escalators and heads to a fixed check-in point. The user profile is [Adult, Effort-Saving-Motion, Visitor, 0.5m].

In Schiphol Airport a priority order is applied: the first one is the Floor strategy and the second is Central HC criterion for travellers who prefer VUs close to them. From a space on the ground floor to another on the top floor, six logical paths are computed according to the Floor strategy (Figure 7.3a to f). Two elevators are involved for the six paths. In order to decide on one path, the Central HC criterion is applied, and this makes it possible to obtain the final logical path (Figure 7.3f). Generally Central HC helps to reduce the number of Floor strategy paths, and ensures the traveller moves horizontally in central regions. The central regions normally include salient signs and they are pivots in the building, which provide more information for guidance. The corresponding geometric path is then computed for the traveller shown in Figure 7.3g. Figure 7.3h presents all the geometric paths related to the other five logical paths.
Indoor Semantic Modelling for Routing
(d) Logical Path 4

(e) Logical Path 5

(f) Final Path
FIGURE 7.3  Logical and geometric paths of option C3.1 in Schiphol Airport. The logical paths are computed with the Floor strategy. (a) Path 1; (b) Path 2; (c) Path 3; (d) Path 4; (e) Path 5; (f) Final logical path (in red) selected by Central HC criterion; (g) The geometric path based on the final logical path; and (h) The other geometric paths (in orange) derived on the unselected logical paths.
By comparing figures 7.2 and 7.3, it is noticeable that there are fewer logical paths in OTB than in Schiphol Airport, because OTB has far fewer VUs, which limits the options for switching floors. Generally, more VUs in a building may increase the number of paths for the Floor strategy (see Figure 7.3). This is an indication that the Floor Strategy is more suitable to be applied to cases that contain many VUs. Otherwise, a common routing approach is enough to provide paths crossing two floors (e.g., the shortest path).

However, the resulting geometric path in Figure 7.3g is longer than the shortest geometric path derived from another logical path (Path 5 in Figure 7.3e). Table 7.3 presents the length of the six geometric paths; the final path is 162.21m long but the shortest one is 159.35m (Unselected Path 5). Paths computed by the two-level routing are different to the shortest path in a complete geometric network, because distance is not the primary consideration. This routing result shows that a geometric path of two-level routing is generally longer than the shortest path.

### Table 7.3

<table>
<thead>
<tr>
<th>Path</th>
<th>Length (m)</th>
</tr>
</thead>
<tbody>
<tr>
<td>The Final Path</td>
<td>162.21</td>
</tr>
<tr>
<td>Unselected Path 1</td>
<td>189.61</td>
</tr>
<tr>
<td>Unselected Path 2</td>
<td>224.16</td>
</tr>
<tr>
<td>Unselected Path 3</td>
<td>227.26</td>
</tr>
<tr>
<td>Unselected Path 4</td>
<td>185.03</td>
</tr>
<tr>
<td>Unselected Path 5</td>
<td>159.35</td>
</tr>
</tbody>
</table>

§ 7.2.2 Routing with ordered SOIs and one size

For option C3.6 in OTB, the user profile is [Adult, Walking, Visitor, 0.5m] (see Table 7.2). One can consider the following scenario: a visitor wants to meet two persons in different parts of the building. The user wants to specify SOIs and POIs as the first person’s office and his desk, and set the second person’s office and his desk as the target. For this case the SOI strategy is applied to compute a logical path followed by computation of a geometric path.

The start space is on the third floor and the target space is on the second floor (Figure 7.4a). A POI and the related SOI are set on the first floor. The user receives one logical path consisting of two parts: the first half is from the start space to the SOI, and the second half is from the SOI to the target space (Figure 7.4a). Geometric paths related to the two parts are computed and illustrated (Figure 7.4b). Because the building is regularly shaped, the sequence of the highlighted spaces (the logical path) roughly reflects the shape of the geometric path (Figure 7.4). In such cases the user may not need the geometric path since she/he can follow the logical path.

In Schiphol Airport, the scenario of option C3.6 can be associated with the following scenario: a traveller wants to get some food in a cafeteria, before she/he walks to a check-in point. The traveller’s profile is [Adult, Walking, Visitor, 0.6m]. The SOI strategy is implemented for this scenario where SOIs are the cafeteria and the space of a check-in point (the target) (see Figure 7.5).

The start space, the SOI and the target space are used to compute logical paths with the SOI strategy. This results in one logical path only and the related geometric path.
FIGURE 7.4 Option C3.6 in OTB. (a) The logical path. Black arrows indicate the direction of motion, and (b) Corresponding geometric path. The first half is in red to the POI, and the second half in dark green.
FIGURE 7.5 Option C3.6 in Schiphol Airport. (a) The logical path. Direction is indicated by black arrows; and (b) Corresponding geometric path. The first half is in red, and the second half in dark green.
The shapes of the logical path (highlighted spaces) and the geometric path in the airport are not quite the same. The spaces in this building are wider and more geometrically complex (i.e., not square rooms and corridors with small sizes), and they contain many obstacles that disturb the geometric path. This is an indication that a geometric path can be very useful in large spaces.

By comparing figures 7.4 and 7.5, one can find that spaces in the logical path of option C3.6 depend on the ordered SOIs. Though OTB is much smaller than Schiphol Airport, the logical path in OTB (Figure 7.4) includes more spaces (twenty three) than those (nine) of Schiphol Airport (Figure 7.5). In such cases, geometric paths have to be computed to determine the accurate length regarding their related logical paths. More spaces on the logical path might imply that the building is subdivided in detail and that the geometric path may not be very different to the logical path.

§ 7.2.3 Routing with ordered SOIs and changing sizes

An interesting option is C3.3 which involves ordered SOIs and different sizes of the same user. Changes of user size has not been considered in research so far. For the testing purpose, in OTB and Schiphol Airport SOIs are specified far from each other, which increases the number of spaces covered by computed logical paths.

Option C3.3 is implemented in OTB with the user profile [Adult, With-Wheeled-Devices, Maintenance Worker, (0.8m, 0.5m)]. Suppose a maintenance worker needs to check electrical wiring in the building, she/he drives an inspection vehicle and finishes the check, then the worker walks down to the ground floor and leaves the building. Thus the routing involves two user sizes: the first one is 0.8m (with the vehicle) and the other one (alone) is 0.5 m. The SOI strategy is also used in option C3.3 since the location of the user size change refers to a SOI. Note, OTB contains almost no obstacles, and therefore the geometric path looks direct and simple (Figure 7.6).

A possible scenario for option C3.3 in Schiphol Airport could be: an airport staff member who takes care of transportation of customers’ suitcases with a trolley and needs to move personally checked-in oversized baggage to different ports. The staff’s profile

(180) Indoor Semantic Modelling for Routing
is [Adult, With-Wheeled-Devices, Administrator, (0.5m, 1.2m)]. Therefore, the staff needs elevators and her/his user size is with respect to the oversized baggage. The staff’s size changes from 0.5m to 1.2m. The SOI strategy is applied to this case. The first half of the geometric path is for size 0.5m (the staff), and the second part is for size 1.2m (the staff with devices, see Figure 7.7).

![Diagram of geometric path](image)

**FIGURE 7.7** Option C3.3 in Schiphol Airport. (a) The geometric path from the start location to the POI for 0.5m (in red); and (b) The geometric path from the POI to the destination for 1.2m (in dark green). The bold polygons represent the newly grouped obstacles.

When a user size changes in a building, the geometric network has to be updated for routing, because edges of the network are changed. In order to investigate the computational load of geometric network generation, computational costs of the two-level routing are compared with routing in a complete geometric network. In option C3.3, when the user size changes at a POI, two-level routing computes the next logical path and group obstacles in the spaces of the logical path according to the new size, and then computes the geometric path. The computational cost on a complete geometric network consists of the following steps: 1) grouping obstacles (see Chapter 5); 2) recreation of a complete geometric network (see Chapter 6); and 3) routing on the new network. The cost of path computation is mostly influenced by grouping obstacles, i.e., more obstacles increase the computational cost.
Table 7.4 lists the comparison of computational costs of the two approaches for the old OTB building and Schiphol Airport. Segment refers to the two parts of a geometric path (see the column Size). The column Total cost shows the entire routing cost. Cost ratio shows the cost ratio of two-level routing to routing in the complete geometric network. In both buildings, the two-level routing approach is significantly faster (1.59s and 1.19s) for a quick change of the user size. If the whole geometric network is updated to compute the geometric path, the process is prolonged (10.61s and 4.73s). The creation of a complete geometric network of OTB (4.73s) is shorter than that of the airport (10.61s), because OTB contains few obstacles. The geometric network creation in OTB saves the cost of grouping obstacles and obtains a simpler geometric network.

TABLE 7.4 The cost to compute geometric paths regarding routing option C3.3 for Schiphol Airport and OTB. The unit is second(s).

<table>
<thead>
<tr>
<th>Building</th>
<th>Segment</th>
<th>Cost of two-level routing (s)</th>
<th>Total cost (s)</th>
<th>Cost of routing with complete geometric networks (s)</th>
<th>Total cost (s)</th>
<th>Size (m)</th>
<th>Cost ratio (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Schiphol</td>
<td>Space 23 - 14</td>
<td>1.13</td>
<td>1.59</td>
<td>7.498</td>
<td>10.61</td>
<td>0.5</td>
<td>14.99</td>
</tr>
<tr>
<td></td>
<td>Space 14 - 35</td>
<td>0.458</td>
<td></td>
<td>3.108</td>
<td></td>
<td>1.2</td>
<td></td>
</tr>
<tr>
<td>OTB building</td>
<td>Space 117 - 31</td>
<td>0.934</td>
<td>1.19</td>
<td>2.506</td>
<td>4.73</td>
<td>0.8</td>
<td>25.16</td>
</tr>
<tr>
<td></td>
<td>Space 31 - 0</td>
<td>0.253</td>
<td></td>
<td>2.226</td>
<td></td>
<td>0.5</td>
<td></td>
</tr>
</tbody>
</table>

As shown in Table 7.4, it is not suitable to obtain geometric paths by updating the complete geometric network, when a user size frequently changes (e.g., keeps transporting different luggage in the airport). However, if one knows the range of size changes in advance (e.g., from 0.4 m to 1.0 m with 0.1 m intervals), the pre-computed complete geometric networks can still save time especially for frequent routing requests.

The OTB building contains only two obstacles and thus they have little influence on a geometric path. Therefore, the geometric networks for sizes 0.5 m and 0.8 m have no obvious difference. In such an environment, user size change has little influence as long as the maximum user size fits the openings (e.g., the user can pass through doors). Comparing the cost ratio (the column Cost ratio in Table 7.4) of the paths in the two buildings, two-level routing in Schiphol Airport is more efficient (14.99%) than for OTB (25.16%). As mentioned before, this is because OTB has few obstacles, and thus less time is needed to process obstacles and compute the complete geometric network.

Changing user size is a kind of real-time information. Although it is not reasonable to enumerate and store geometric networks for all possible user sizes, in some special environments (e.g., an employee in a factory manipulating different devices) one may frequently encounter a situation where there needs to be an update or an ad-hoc computation for geometric networks and paths. When geometric networks are available for different user sizes, the test in Schiphol Airport still shows that the two-level routing is a more economic method than routing on the complete geometric network (see Table 7.4). Considering the computational cost reduction is more efficient in Schiphol Airport (14.99%) (see Table 7.4), it can be qualitatively concluded that the two-level routing approach can achieve a better performance in complex buildings.
TABLE 7.5 Involved points of geometric networks with regard to option C3.3 in Schiphol Airport and OTB.

<table>
<thead>
<tr>
<th>Type</th>
<th>First Point</th>
<th>Partial geometric network</th>
<th>Complete geometric network</th>
<th>Ratio (%)</th>
<th>Second Point</th>
<th>Partial geometric network</th>
<th>Complete geometric network</th>
<th>Ratio (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Point</td>
<td>Schiphol Airport</td>
<td></td>
<td></td>
<td></td>
<td>OTB</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>1945</td>
<td>5118</td>
<td>38.00</td>
<td>612</td>
<td>2809</td>
<td>612</td>
<td>21.79</td>
</tr>
<tr>
<td></td>
<td></td>
<td>746</td>
<td>5170</td>
<td>14.43</td>
<td>743</td>
<td>2854</td>
<td>743</td>
<td>26.03</td>
</tr>
</tbody>
</table>

In order to evaluate the data amount of two-level routing, statistics (see Table 7.5) are given on the number of points used for geometric networks in option C3.3. The sub-column ‘Partial geometric network’ refers to geometric networks derived by two-level routing. The sub-column ‘Ratio’ shows the ratio of the point numbers of two-level routing to those of related complete geometric networks. As introduced in Chapter 3, geometric nodes are indoor transfer locations (e.g., door centres) and POIs, and a geometric edge is the shortest path between two nodes as a polyline. Besides the geometric nodes, each geometric edge may contain intermediate points such as obstacle vertices and corners of spaces. In addition, points of INSM refer to its polygons of NavigableUnit (NU), Opening (OPN) and Obstacle(OBS). For example, a rectangular NU is represented by a rectangle of 4 points.

In Schiphol Airport and OTB, all the points are counted for geometric networks derived by two-level routing, the complete geometric network, and the INSM (see Figure 7.8). In the statistics of Schiphol Airport, the first part represents the largest ratio of points (38.00%) of the complete geometric network for size 0.5m, and the second part has only 14.43% points of the complete geometric network for size 1.2m (see Table 7.5). In OTB, the larger ratio of points (26.03%) is in the second part. This indicates that two-level routing needs less half storage loads compared to the complete geometric network.

By comparing the points of the complete geometric network and its INSM, one can find that the complete geometric network has far more points than the INSM (Figure 7.8). Regarding the two complete geometric networks (for two different sizes) in Schiphol Airport, both the point numbers (5118 and 5170) are much more than that of the INSM (1947). In the old OTB building, the point numbers (2809 and 2854) of both the complete networks are considerably more than that of the INSM (1714). This fact shows that the complete geometric network carries the largest data amount, which needs to be considered when storage is sensitive (e.g., a mobile application).

Compared to OTB, the complete geometric network of Schiphol Airport has far more points (5118 and 5170) (Figure 7.8), which indicates the interior of OTB is less complex (e.g., fewer obstacles) than Schiphol Airport. The point numbers of two-level routing in both the buildings also support this proposition: Schiphol Airport needs more points (1945 and 746) for two-level routing.
FIGURE 7.8  Statistics of the involved points for Schiphol Airport and OTB. (a) Points with respect to Schiphol Airport; and (b) Points with respect to OTB.
§ 7.3 Mobile application

This section presents a mock-up of two-level routing in a mobile application developed with Bentley Systems Mobile SDKs. It refers to Case 2 of Table 7.1. The purpose of the development is to conduct two-level routing in mobile devices in a more convenient way. This test is used to demonstrate the feasibility of the two-level routing approach. In this mobile application, the INSM, the logical network, and a derived complete geometric network are stored in an i-model file with the internal database ECDB [Sys16c], i.e., a SQLite [Con16] database. ’EC’ represents the term 'business data' which is a Bentley information modelling system, and ‘ECDB’ is the API to access the EC data. In my implementation, the ECDB file is stored in the client. Thus users can obtain logical and geometric paths even when they are disconnected from the internet. A set of SDKs named Graphite [Sys16c] is adopted for implementation. The two-level routing function is developed in the Navigator Mobile [Sys16c]. Navigator Mobile provides basic interface and functionality to import and visualize graphic data.

ECDB files are designed for applications that work with non-graphic data. ‘EC’ represents Bentley’s information modelling system that involves self-describing non-graphic data. An ECSchema defines the data model for an ECDB file. The ECSchema is represented by a XML-based file. An ECSchema consists of ECClass and ERCRelatio nClass, and an ECClass contains ECProperty as attributes (see Figure 7.9). Relationships within ECClass are described by ERCRelationshipClass. In general, ECClass represents tables of the database and ECProperties are columns of the ECClass, while ERCRelationshipClasses performs similar to link tables. In addition, a generic concept is used to refer to the primary key and the foreign keys of an instance, i.e., ECInstanceId. ECInstanceId is the equivalent concept of a primary key. Each ECClass has a built-in ECProperty called ECInstanceId which has no need to be explicitly defined in the ECSchema.

![FIGURE 7.9](image) A snippet of the ECSchema. Structures of ECClass, ERCRelationshipClass and ECPROPERTY are shown in the snippet. ECPROPERTY are organized under the ECClass, ERCRelationshipClass tags.
Testing data is prepared for the Navigator Mobile which originates from the Residence building in IFC format (see Chapter 6). Firstly, the house model is mapped into INSM. Secondly, based on the INSM, the logical network and a complete geometric network for users of size 0.5m are computed by the desktop application. Thirdly, the floor plans with the logical and the geometric networks are published as an i-model file by using MicroStation V8i, which adds the different data in a package of i-model. The published i-model contains graphic (the INSM and/or 3D building model) and non-graphic data (i.e., ECSchema). The i-model of the building is the input for the Navigator Mobile. Finally the i-model is loaded in the Navigator Mobile. Through the above steps, Case 2 in Table 7.1 is implemented (i.e., thick client in mobile application).

The interface of the Navigator Mobile is presented in Figure 7.10 where the ‘Browse’ option is for selecting i-model files. The first step is to select and load an i-model from a local disk or servers. Here my local i-model is loaded with a 3D building model of Residence. Snapshots of the building and the INSM are shown in Figure 7.10. In Navigator Mobile, developed functions mainly include options to select, view and measure building geometry. The 3D building and the INSM are stored in different layers and can be selected for visualization in Navigator Mobile. This research adds new functions for two-level routing.

Based on the Navigator Mobile, functions are added for logical/geometric network generation and two-level routing. Specifically three functions are developed: 1) to select the start and destination by identifying POIs on a plan view of a navigationally enabled i-model; 2) to compute logical paths and geometric paths with the i-model; and 3) to visualize logical and geometric paths. After loading the i-model data in the Navigator Mobile, a test is conducted with the two-level routing approach.

Figure 7.11 presents the structure of designed ECSchemas of logical and geometric networks. The conceptual models of the logical and geometric networks are implemented as presented in Chapter 3. As mentioned above, the ECSchema format provides two types of classes: ECClass and ECRelationshipClass. ECClasses denote the nodes of a logical/geometric network, and ECRelationshipClasses are used to represent the edges of a logical/geometric network. The ECRelationshipClasses clarify the relationships of two nodes (i.e., connectivity or a geometric edge) and the source and the target of a relationship. For a logical network, a class Node is created with two attributes, Name and SpaceType (Figure 7.11a). The Name helps a user to understand indoor spaces, and the SpaceType uses values to represent space type according to the INSM semantics of a space. For example, different numbers (from 1 to 6) are set to HorizontalConnector(HC), VerticalConnector(VC), END, Stair, Elevator and Escalator, respectively. An ECRelationshipClass Edge is created without attributes.
Realization of two-level routing
FIGURE 7.10 Navigator Mobile application. (a) The start interface; (b) The Residence building; and (c) The INSM.
FIGURE 7.11 Schemas of logical and geometric networks. (a) The schema of logical networks. It contains the ECClass Node and ECRelationshipClass Edge; and (b) The schema of geometric networks.
The ECSchema of geometric networks corresponds to the conceptual model of the geometric network (see Figure 3.16). Two ECSchema classes OpeningNode and POI represent geometric nodes, and they include the same attributes of the classes OpeningNode and PointOfInterest in the conceptual model of the geometric network (see Figure 3.16). Here the conceptual models of logical and geometric networks are implemented by the ECSchema. This implementation is illustrated with the UML Physical Model (Figure 7.12 and 7.13). Note that the class Edge has two attributes StartNodeId and TargetNodeId to refer to a directed logical edge (Figure 7.12b). Two connected spaces are mapped to two directed edges when both the directions are accessible.

**FIGURE 7.12** Data model of mobile implementation of the logical network. (a) The conceptual data model; and (b) The physical model.
FIGURE 7.13 Data model of mobile implementation of the geometric network. (a) The conceptual data model, and (b) The physical data model.
Four relationship classes *Opening2Opening*, *Opening2POI*, *POI2Opening* and *POI2POI* represent geometric edges (many to many relationships, see Figure 7.13b). The GeometricEdge’s attributes *NodeType*, *TargetNodeType*, *StartNodeId* and *TargetNodeId* in the conceptual model are implicitly referred to by the four relationship classes. From the name of the relationship classes one can immediately know the types of start and target nodes (POI or Opening). The four relationship classes include three attributes: *Points*, *Length*, and *SpaceId* (Figure 7.13b). The *Points* stores the coordinates of a geometric edge (a polyline). The *Length* stores the distance of the geometric edge, and the *SpaceId* indicates the space of the geometric edge. The geometric network is a directed graph and paths of both directions between two geometric nodes are referred to by *StartNodeId* and *TargetNodeId*.

As a demonstration, option C3.1 is applied (constant user size without SOI/POI) in this mobile application. The user profile is set as [Adult, Walking, Visitor, 0.5m], which indicates that the Minimum NU criterion is applied for the logical path and that the stored geometric network fits the size 0.5m. Figure 7.14 illustrates the prototype of the mobile application. Coordinates are assigned to the logical network so that the nodes are embedded in each NU. The start/target locations of a user are selected by identifying two POIs in the *i-model*.

Figure 7.14c presents the complete geometric network in all spaces. Based on the computed logical path on the Minimum NU criterion, option C3.1 is conducted to obtain the geometric path between the start and target locations (see Figure 7.14d).
Start NU

Logical Network

Logical Node

Target NU

Geometric Network

(b)

Realization of two-level routing
FIGURE 7.14 Demonstration of two-level routing in mobile application. (a) The Residence building; (b) The logical network, represented by a geometric network (in orange) containing both space connectivity and vertex coordinates; (c) The complete geometric network (in white); and (d) The geometric path (in green). The green bars are door centres.
§ 7.4 Analysis of the tests

This section analyses all the results presented in sections 7.2 and 7.3, and provides more details on improving the two-level routing approach and the future development of two-level routing.

Two-level routing is based on an INSM with two derived models – the logical network and the geometric network. The INSM contains the geometry and semantics of indoor spaces. The logical network represents the connectivity among indoor spaces. One can compute a complete logical network, and sub logical networks can be retrieved from it (e.g., according to attributes) on the selected spaces for different users (e.g., visitors and maintenance workers). Given a user size, there is one complete geometric network for all indoor spaces but it is not necessary to compute it for routing. Lots of sub geometric networks can be derived in real-time according to specified spaces and different user sizes. For example, geometric networks are updated (e.g., in the public area for the two-level routing) when a user size or obstacles in the airport change.

Considering different combinations of the logical network and the geometric network, the two-level routing approach is flexible to provide different paths. For example, the logical path can be conveyed to users verbally by describing space names, while the geometric paths in one or more spaces can give a geometrically accurate path for a specific size.

The result of the first experiment in Section 7.2 shows that more logical paths can be found in a large and complex building which contains many VUs, such as the building of Schiphol Airport. The existence of multiple paths also depends on the location of the start and target spaces. Even in complex building with many VUs, it is still possible to obtain just one Floor strategy path to certain target spaces. However, more VUs in a building do indeed increase the vertical transfer opportunity, which may generate more Floor strategy paths from some spaces (see Figure 7.3). The number of Floor strategy paths in Schiphol Airport and OTB indicates that choices of logical paths are more definite in a simple building where multiple routing criteria are not needed. In contrast, routing in Schiphol Airport needs ordered routing criteria to reduce logical paths. This comparison manifests that routing with multiple criteria is more suitable for complex buildings.

Generally the geometric paths derived from the two-level routing can be longer than the shortest path in a complete geometric network. The result in Section 7.2.1 has shown that two-level routing does not give the first priority to the distance. According to a user’s needs, the routing can result in proper logical paths and longer geometric paths.

The test with option C3.6 (with ordered SOIs/POIs and a constant user size) shows that simple regular buildings most probably contain similar shapes of the computed logical path and the corresponding geometric path. In such cases, a user can recognize and follow these spaces without the geometric path. In contrast, complex irregular buildings, often influenced by the irregular geometry of spaces and indoor obstacles, require the computation of geometric paths. Therefore, it can be concluded that two-level routing is more efficient in complex buildings.
The space number of logical paths is also compared in simple and complex buildings. The logical path in a simple regular building may include more spaces than in a complex building. This result indicates that the length of two different logical paths could not be precisely compared by their related space numbers. This is because a space can be large or very small, and thus geometric paths in two spaces can vary considerably. A logical path with a few spaces may correspond to a long geometric path (e.g., one long corridor), but another logical path with a number of small spaces can derive a shorter geometric path.

The experiment of option C3.3 (with ordered SOIs/POIs and different user sizes) demonstrates that two-level routing can handle the changes of a user size more efficiently, compared to routing with a complete geometric network. In general, two-level routing needs less cost for routing when user sizes change. It is advised to compute geometric paths by conducting two-level routing when the user size frequently changes. Tests in the two buildings show that two-level routing is suitable for buildings with many obstacles. In the airport, the complex building, computational cost is much less than that with a complete network (see Table 7.4).

Generally some issues have not yet been addressed in two-level routing, which are:

- **Multiple resulting logical paths.** Routing in complex buildings possibly derives more than one logical path even with ordered multiple criteria. In such cases, this research would leave all the resulting logical paths to a user. In the future, a method could be developed to ensure the only logical path, which can save a user’s effort to decide the path to be followed. For example, besides using multiple ordered criteria, different criteria can be weighed and organized in an objective function for optimization.

- **The relationship between user profiles and routing criteria.** This research designs a simple user profile and assumes the connections between different profiles and ordered routing criteria. To gain more realistic routing results, it is necessary to investigate genuine user needs by questionnaires. Then a more reasonable relationship would be provided for different users.

- **Customized geometric paths.** So far user needs on geometric paths have not been considered. These paths are computed with the Dijkstra algorithm [Dij59] in a geometric network. In the next steps, semantics in geometric networks could be used to realize user requests. For example, geometric nodes can represent a door, a window, a corner, or other POIs. Considering a user’s preferences on geometric paths, routing criteria can be developed in geometric networks (e.g., the closest door or the minimum corners).

In this chapter, the two-level routing mock-up in the mobile application is only an initial experiment. This application is developed as a thick client (i.e., Case 2 in Table 7.1), and all the computation are conducted on the client without the internet. It demonstrates that two-level routing can be added in a mobile application. It is currently only for routing with one user size. Though theoretically it can store more than one complete geometric network for different user sizes, this may challenge the computational ability of mobile devices with a larger dataset. An alternative may be labelling the maximum size for each edge in the complete geometric network. Then paths can be computed for different given user sizes.
As future work, it is recommended to experiment with other mobile options of the two-level routing in Table 7.1. The current implementation can be improved by using servers to conduct two-level routing. The two implemented thick client cases (Case 1 and Case 2) provide two-level routing only on the client side. Based on the Client-Server structure, with limited effort one can implement the mobile applications of cases 3, 4 and 5 (see Table 7.1). The three cases represent thin clients where two-level routing is conducted on the server side. These thin client applications can provide a user with an updated INSM and related logical and geometric paths, which support data synchronization with clients. The two-level routing can be moved to a server, and then the mobile application is used as a visualization tool. The INSM and the logical network are stored in the server, and geometric networks can be either pre-computed and stored (cases 3 and 4), or derived ‘on the fly’ (Case 5).

§ 7.5 Summary

In this chapter, five cases have been proposed according to the methods of model creation (pre-computed or ‘on-the-fly’), model storage (on clients or servers) and implementation platform (desktop or mobile) (see Table 7.1). The first case is for desktop application which is a thick client application (all data and computation only on the desktop client without server). The second is also a thick client application (without server) but implemented for mobile application. The third case includes all the data stored on the server side, but the mobile client also stores a copy of the INSM and the logical network. In the fourth case, all the data is stored in the server, and none in the mobile client. The fifth case only stores the INSM and the logical network on the server side, and none in the mobile client.

The first two cases are implemented as a desktop and a mobile application, respectively. Tests on the two applications answer the following research sub-question:

6. How are the new proposed user-related paths implemented and applied to realistic cases?

In the desktop application, user-related paths are presented with two-level routing in the simple OTB building and the complex Schiphol Airport, which shows the generation of logical and geometric paths according to user preferences on logical paths, ordered SOIs and POIs, and user sizes. The logical path excludes unrelated spaces, and then the geometric network is created in the selected spaces for routing. In Section 7.2 three routing options are applied to the desktop application, i.e., option C3.1 (constant user size without SOI/POI), option C3.6 (constant user size with ordered SOIs/POIs) and option C3.3 (changing user sizes with ordered SOIs/POIs). From the tests, one can conclude that routing with multiple criteria on the logical network is more suitable for a complex building, and that two-level routing can process changes of user size more efficiently compared to using the complete geometric network.

Section 7.3 presents the development of the mobile application with Bentley Systems special tools such as i-model and ECDB (Bentley’s API to access EC data) [Sys16e]. Option C3.1 for two-level routing is implemented in the Residence building to obtain a
geometric path, which demonstrates the feasibility of this thick client application for users without an internet connection.

Section 7.4 discusses the options not addressed in this implementation, including the reduction of multiple resulting logical paths, validation of the relationship between user profiles and routing criteria, and consideration of the semantics of geometric nodes for specific user needs. These topics will be left for future work. In the following chapter conclusions of this research will be drawn.
8 Discussions and conclusions

This final chapter summarizes the main findings of this PhD research, and proposes future work to continue the research. In Section 8.1, the answers to the research questions (Chapter 1) are provided. Section 8.2 presents an assessment of the proposed two-level routing approach. Section 8.3 lists recommended research topics as the future work of this PhD research.

§ 8.1 Outlook on this research

Complex buildings most often contain many indoor spaces with irregular shapes and indoor objects, which requires user-related paths in addition to the shortest path. Human users have different preferences for paths through indoor spaces and locations. The semantics of indoor spaces provides a way to measure the fitness of spaces to a user. In addition, a user’s size needs to be considered to find an accessible detailed path. Thus a user embodying user preferences and sizes should be applied to indoor routing. The resulting paths are user-related and thus differ in terms of users. All the above topics are investigated in this thesis under the following main research question:

- What indoor routing approach can provide accessible paths according to human user preferences by using the semantics of indoor spaces, in addition to using building topology and geometry?

I have devised and tested an innovative approach for indoor navigation named the two-level routing approach. This approach can generate paths based on the proposed Indoor Navigation Space Model (INSM) semantics of indoor spaces and space geometry. The approach can also adjust indoor routing according to user needs, such as passing through ordered SOIs, POIs and obstacle-avoidance. The resulting paths are adaptable to users with varying sizes.

The sub-questions are repeated and the answers are elaborated in the text below.

1. What kind of information, data models and routing algorithms has been used and developed so far, and what are their limitations for large complex buildings? (Chapter 2)

To complete the indoor routing task, one needs the semantics, topology, and geometry of a building. Building models (Section 2.1) as data input are required, such as CAD files of floor plans, standard data of city models (i.e., CityGML) and standard data of buildings (i.e., Building Information Model (BIM)). However, CAD files always lack the semantics of indoor spaces and contain very primitive geometry (e.g., lines). Semantic models of CityGML [GKNH12] and BIM [IAI16] contain abundant space semantics and accurate geometry of the spaces (e.g., 2D surfaces or 3D solid). However, the semantic models are not specifically for indoor navigation and cannot be directly used for navigation network generation. Another semantic model named IndoorGML focuses on indoor spaces and their connectivity for navigation networks [LLZ+14]. But the network
of navigable spaces (i.e., its Navigation module) is not compulsory for IndoorGML, and the navigation networks have to be derived from building models.

Space semantics should be considered in the generation of navigation networks. There are many ontologies which define semantics in different ways [TAKH06, DGK09a, KG10, GZ11a]. In order to apply any of the ontologies, one needs to consider the subdivision approach of building (e.g., based on the structure or functions of spaces). However, the method of subdivision is not clear, because the semantics of the reported ontologies are either too general or too detailed for different types of building.

Navigation models (i.e., 2D/3D networks or grids) can be generated from the building models for path computation (see Section 2.2). In general, there is no standard navigation model for every case of indoor navigation. Therefore, a navigation model needs to be selected in a specific context. For indoor pedestrian navigation, navigation networks are extensively used. For instance, the Straight-Medial Axis Transformation (S-MAT) modelling method [EE99, CL09] is frequently applied to indoor obstacle-abundance scenarios to get a medial axis network of spaces. Though Mortari et al. [MZLC14] propose a similar network model to improve the S-MAT methods to incorporate obstacle, such networks are not flexible enough to handle indoor obstacle changes. In contrast, Visibility Graph (VG) [Lat91, dBCvKO08] does not follow the shape of indoor spaces, but provides direct paths among locations. Some research of indoor pedestrian navigation employs VG-based methods for indoor obstacle scenarios, and the results reflecting the VG-based methods can better deal with indoor static obstacles [HBK+10, KBH12, SGS12].

Although shortest-path algorithms such as Dijkstra [Dij59] and A* [HNR68] are widely used for indoor routing, many researchers propose some ad-hoc routing methods and routing criteria depending on specific navigation models and user requirements (Section 2.3). Except distance, travel time or number of turns, other non-metric factors are considered (e.g., cognitive similarity, temperature, and visual signs). Some pedestrian-related paths are also defined, such as ‘feasible’ and ‘comfortable’ paths for a wheelchair user [DGK09a], least-effort and least-visible paths [LZLF08, CWSC14]. There are routing methods defined on specific navigation models or structures, such as cactus tree-based routing [WMY07], routing on Octree structure derived from point clouds [RVZ16], etc. Pedestrian wayfinding research (Section 2.4) presents some heuristics about human wayfinding behaviours, which can be adopted to design routing criteria. However, two issues are seldom discussed for indoor pedestrian routing: 1) routing with multiple criteria; 2) dimension/size of pedestrians.

2. What data and navigation model is appropriate to represent the semantics, topology and geometry of indoor spaces? (Chapter 3)

I proposed routing for two types of network, i.e., the logical and geometric networks. As mentioned before, this thesis uses the network model. The logical network sustains the semantics of indoor spaces and their connectivity. The geometric network maintains the accessible edges between geometric nodes for a user, which is derived from the geometry of buildings. This research has shown that separating the semantic and geometry increases the flexibility for indoor routing (Chapters 3-5). According to user preferences, routing results on the logical network can filter space candidates and thus generate detailed paths more efficiently.
I proposed the INSM model to store the building information and to facilitate the derivation of the navigation networks for path computation. Spaces of INSM are linked to nodes and edges of navigation networks, and both logical and geometric networks can be derived from the INSM. Indoor obstacles are also maintained in the INSM, which is important for obstacle-avoidance of indoor navigation.

Specifically, INSM includes abundant semantics of building components based on navigational functionalities. INSM also manages the connectivity and geometry of indoor spaces (see Section 3.2). INSM captures a set of non-overlapping spaces and their semantics according to their functionalities for indoor routing, such as VerticalUnit (VU) and HorizontalConnector (HC) as vertical and horizontal passages for human users. Semantics of INSM are used to design routing criteria applied to indoor routing in the logical network. In addition to semantics, connectivity of spaces should be detected in building data and explicitly retained in the relationship of NavigableUnit (NU) and Opening (OPN). INSM also maintains the geometry of indoor spaces which can be stored as 2D polygons or 3D solids. The geometry of indoor spaces and obstacles are required to support the computation of an accessible area for a user and the generation of accessible geometric networks. In my implementation I used 2D polygons to represent navigable surfaces for pedestrian users. The logical network is created by connectivity of spaces and the semantics contained in INSM, and the geometric network is derived based on the geometry of each space and the obstacles.

**FIGURE 8.1** The INSM model with classes and attributes (same as Figure 3.6 in Chapter 3).

Important attributes of the INSM classes are required as well (see Section 3.2). In general, I need to collect the name, top and bottom heights and the floor of spaces from building data. A NU needs attributes Name, BottomHeight, and TopHeight (Figure 8.1) to describe the space’s name, the lowest height and the top height, respectively. VU, a subclass of NU, indicates its containment status by its aggregation relationship Is-Contained (Figure 8.1) to HorizontalSpace (HS). In addition, the attribute Type refers the specific subtype of the VU (i.e., Stair, Elevator and Escalator). HU, the other subclass
of NU, contains the attribute Type to specify its subtype (i.e., HorizontalConnector–HC, VerticalConnector–VC and End). The aggregation of HU to HS reflects the floor of this HU instance. An OPN has two Boolean attributes IsExisted and IsLocked which indicate the status of the OPN (e.g., virtual or physical, and locked or not), respectively. An OPN also includes the attribute Type to specify its subtypes (Door, Doorway, FacadeWindow, InteriorWindow and MainEntry, see Figure 8.1). OBS has no attributes since it is an occlusion space just to be avoided.

Based on INSM, and its derived models of the logical and the geometric networks, this research proposed an innovative two-level routing approach which adopts different types of indoor space, i.e., general free spaces for the logical network, and openings and SOIs for geometric networks. The logical and the geometric network refer to different types of space. A subdivision of a building results in indoor spaces as rooms. The nodes of a logical network represent the rooms and the edges stand for the connectivity of the rooms. The nodes of a geometric network refer to transition spaces occupied by doors and windows and subspaces (SOIs) inside of these rooms, and the edges represent detailed paths among these geometric nodes. This new two-level routing is different from the most common approach in previous research, i.e., hierarchical graphs. For hierarchical graphs, their nodes on different levels are linked to different subdivision results (e.g., floors, sections, rooms, sub-rooms) that are all general free spaces. In contrast, with the two-level routing approach, a detailed path can be relevant to both general free spaces and transition ones (e.g., doors).

3. What kind of user-related paths can be computed with the semantics, topology and geometry of indoor spaces? (Chapter 3)

A number of user-related paths can be computed according to the proposed approach. My two-level routing is not only about the shortest or fastest path in a building. In Chapter 3 I have proposed seven routing options (C3.1 to C3.7) to flexibly compute user-related paths on both the logical (semantics and topology) and the geometric network (geometry). These options stand for different applications with a single destination (see Section 3.4.3): 1) a user can automatically obtain a logical and a geometric path without specifying any SOI and POI (option C3.1); 2) a user can also obtain a logical path and the related complete geometric path through the specified ordered POIs (option C3.2); 3) considering the changes of a user’s size (e.g., moving with a shopping trolley), logical and geometric paths can be computed through the given SOIs and POIs in order where the user size varies (option C3.3); 4) when users need geometric paths only in specific rooms instead of a complete geometric path, a logical path can be computed for the users to show the sequence of spaces to be passed and highlight the ones regarding geometric paths as SOIs. Then separate geometric paths in these SOIs would be provided for the users (option C3.4); 5) in some routing cases, a user has to pass through certain spaces in a specific order. Thus, the user would get a logical path through the given spaces sequentially, and the related complete geometric path; 6) in other cases, a user may not only need a logical path through the specified SOIs in order, but also ask for the complete geometric path through the given POIs inside each SOI in order (option C3.6); and 7) if a user finds the computed logical path is not satisfactory, she/he can request a re-computation by assigning ordered SOIs (option C3.7). As a result, the recomputed logical path would cross the SOIs sequentially, and the complete geometric path would be updated as well. In each of the above routing options, the resulting geometric path avoids indoor static obstacles, and it is an accessible path for the user according to the given user size.
4. **What kind of routing criteria can be built (or specified) by using the semantics of indoor spaces? (Chapter 4)**

Six routing criteria and three strategies are designed for routing on the logical network regarding the INSM semantics (see Chapter 4). The criteria are *Minimum Navigable-Unit (NU)*, *Minimum HorizontalConnector (HC)*, *HorizontalConnector (HC) Prior*, *Minimum VerticalUnit (VU)*, *VerticalUnit (VU) Prior* and *Central HorizontalConnector (HC)*. *Minimum NU* is favourable for users who need the fewest spaces to be passed. *Minimum HC* benefits users who want to horizontally cross as few HCs as possible. *HC Prior* is suitable for users who prefer HC nodes and also move as directly as possible to the target. *Minimum VU* has benefits for short vertical movements, while *VU Prior* is for users who prefer VU nodes (e.g., always trying to change floors) and also move as direct as possible to the target. *Central HC* helps users to find paths preserving the high level of accumulated centrality of HCs. With these criteria this thesis investigated different minimization processes on node/edge weights of the logical network, which derives logical paths represented by spaces defined in the INSM model.

In the logical network, I also realize three routing strategies (heuristics of wayfinding): the *floor*, *flat location* and *SOI* strategy (see Section 4.2), by using the *minimum NU* criterion. The *floor* strategy gives priority to movement on the same floor until the location is horizontally close to the destination, while the *flat location* strategy prioritizes arriving at the destination floor and then leads the user horizontally to the destination. The *SOI* strategy ensures a user passes assigned *Spaces of Interest (SOI)* sequentially, which benefits the wayfinding process with high salience areas (i.e., the SOIs).

Tests (see Chapter 6) have presented the routing results on logical routing with four case studies, i.e., a residential building, the Museum of Fine Arts (MFA), Schiphol Airport, and the old OTB building (see Section 6.3.1). The case of the residential building shows that there are no multiple paths between two spaces due to the simple structure of the logical network. The cases of MFA, the old OTB building and Schiphol Airport show that multiple logical paths do exist among different spaces. The routing tests in the MFA and Schiphol Airport result in multiple logical paths, and in both cases the candidate paths can be reduced by using user profiles which refer to a priority order of routing criteria.

5. **Which approach should be used to compute the exact geometric description of accessible paths according to the size of a user? (Chapter 5)**

I proposed a new method to compute accessible geometric paths for a user in a space to avoid obstacles (see Chapter 5). According to the user size, this method consists of grouping obstacles, generating new boundaries of the obstacles, and creating the accessible geometric network for the user. This method allows geometric networks to be automatically derived for individual users with different sizes. A part of the nodes of the geometric networks represent doors which are fixed in the building.

The proposed geometric routing regarding user sizes was tested with three case studies (Section 6.3.2): 1) routing for two groups of users with two distinct sizes in a space; 2) routing for the same user with changed sizes in a space (e.g., handling a device) and 3) generating the geometric network in a given sequence of spaces. The first case showed that obstacles largely change accessible geometric paths for a user, even with small changes of the user size. The second case shows that geometric paths are computed...
‘on the fly’, considering the current user size and changes in the next steps. Different segments referring to distinct sizes are combined into the final geometric path. The third case shows how for a user with a constant size, a geometric network is formed by putting together the subnetworks in four spaces. In other words, this geometric network relies on the selected spaces. This is the way to obtain a complete geometric network in specified spaces for the given user size. In my test the shortest path given by a specific logical path is computed on this geometric network for the user to pass through these specified POIs. All three of the tests manifest how accessible paths can be computed for any user, and how a related geometric network or geometric path can be computed in given spaces.

6. How are the new proposed user-related paths implemented and applied to realistic cases? (Chapters 6 & 7)

I proposed five implementation cases according to the methods of model creation (pre-computed or ‘on-the-fly’) and model storage (on clients or servers). The first one is to pre-compute the INSM and logical network and generate the geometric network ‘on-the-fly’, and the two-level approach is conducted on a desktop device. All the models are stored on the client side. In this case, path computations can be completely at the client side, which makes it a standalone application (i.e., no server support); in the second case, all the models (INSM, logical and geometric networks) are pre-computed and stored on the client, which is implemented in a mobile application. This case is also a standalone application referring to a mobile device without the internet. A user can compute both logical and geometric paths on the device; the third case includes all the models pre-computed and stored on the server side, and the client also stores the copy of the INSM and logical network. This case is designed for a mobile application where a user can request paths from the server, or independently compute logical paths on the mobile device and follow them when the internet is not available; in the fourth case, all the models are pre-computed and stored on the server, and none by the client. This case refers to a lightweight mobile application where a user just needs to send path requests via the mobile device and receive derived paths from the server; the fifth case only pre-computes and stores the INSM and logical network on the server side, and none at the client. This case is also designed for lightweight mobile applications where the client (the mobile device) sends requests and receives paths from the server. In contrast to the fourth case, in the fifth case the server would compute geometric networks and paths ‘on-the-fly’.

I implement the first and the second cases on desktop and mobile applications, respectively. For the first case, I illustrate routing results of the options C3.1, C3.3, and C3.6 (see Chapter 7) in the desktop application. The routing options are conducted in the old OTB building and at Schiphol Airport, representing a simple and a complex building, respectively. The results of routing option C3.1 in the two buildings show that the Floor Strategy fits for the cases of Schiphol Airport including many VUs for vertical motions, but not for the old OTB building which has much fewer VUs, which limits the options for switching floors. Then the Floor Strategy from different start spaces on the same floor tends to derive similar paths. The results of option C3.6 in the two buildings show that it is difficult to distinguish the accurate distance of two logical paths by their space numbers. By comparing the shapes of the logical path (indicated by spaces) and the geometric path in the two buildings, I found the path shapes of the two types are similar in OTB but not similar in the airport. Compared to the old OTB building, the space shapes of Schiphol are irregular and indoor obstacles curve the geometric
path. Tests with routing option C3.3 in the two buildings show that the cost of path re-computation is influenced by grouping obstacles, when the user size changes at a POI. Two-level routing avoids all obstacles being adopted for grouping and thus is more efficient than recreating the complete geometric network. The cost reduction in the airport is 85.01%, and in the old OTB building it is 74.84%.

For the second case, I presented how the mobile application was developed with Bentley Systems Mobile SDKs. I stored the logical and geometric networks in the Bentley internal database – *ECDb* files. The test building is the residential building in *IFC* format (see Chapter 6). The INSM is generated for the residential building, and also the logical network and the geometric network for 0.5m size. Finally, I publish the three models in *ECDb* files with the geometry of the residential building as an *i-model*, i.e., the data format for *Navigator Mobile* – the Bentley Systems mobile development environment. I visualize the three models in *Navigator Mobile*, and also visualize the computed logical path and geometric path. This test demonstrates that two-level routing can be independently conducted in mobile applications without routing support from a server.

### 8.2 Advantages and further opportunities of the two-level routing approach

The two-level routing approach has three main benefits:

1. The approach is specifically developed and aimed at complex buildings for indoor routing;

2. It is flexible enough to meet the distinct requirements of users for path details. A user can decide to receive indicative logical paths or comprehensive geometric paths, depending on the complexity of a building, and the user can request accessible paths through any SOI and POI in order;

3. The two-level approach can save computational cost by avoiding the recreation of the whole geometric network of a building, in light of different user sizes.

The two-level routing approach includes clear routing criteria and more routing details than hierarchical graph methods. As a modelling method, hierarchical graphs [HD04, JM05, LOS06b, SSO08, YCDN07, RWS11] represent indoor environments according to the cognitive understanding of space hierarchy. These previous studies address this hierarchy construction, but seldom mention routing details on hierarchical graphs, especially for complex buildings.

The first experiment in Section 7.2 shows that the large and complex Schiphol Airport contains multiple logical paths among its spaces due to a number of VUs. The test also manifests how routing with multiple criteria is suitable to be applied to the complex building, which provides fewer logical paths according to a user’s preferences.

Another test with option C3.6 (Section 7.2) is conducted on the simple OTB building. Even without the resulting geometric paths, the computed logical path (a sequence of spaces) is enough to guide users in such a simple building. In a complex building, irregular shapes of spaces and obstacles curve the shape of a geometric path so that only the logical path cannot provide enough details to direct the motion of the user. Based
on the given level of details, the two-level approach can be flexibly adjusted and then delivers the requested path form — logical or geometric paths, or both. The resulting geometric paths are determined by logical paths derived by these proposed routing criteria and user preferences. The geometric path is the shortest path in the geometric network regarding spaces indicated by the logical path.

The last experiment of routing option, C3.3 (see Section 7.2), demonstrates two-level routing can process changes of a user size with less time cost, in contrast to routing on a complete geometric network. In summary, two-level routing may not generate the shortest path in the geometric network of all spaces, but it can flexibly provide logical and geometric paths according to user preferences and sizes, and can adjust the generated paths in a limited time.

The proposed two-level routing approach has the following features:

- Supports routing in different abstraction forms of a building (Chapter 3). The INSM model allows two types of routing network to be derived – pure logical and geometric. The logical network contains topology and semantics of indoor spaces, and the geometric network provides accurate geometry for paths. A consistent navigation model is formed with the two networks, i.e., the conceptual and detailed levels (Chapter 3).

- Supports routing on a logical network and thus assists the generation of a conceptual path for a user in terms of space sequence. Routing criteria are designed based on the INSM semantics of spaces, which can generate logical paths similar to wayfinding results such as minimizing $VU$ or $HC$ (Chapter 4).

- Considers the size of users and results in obstacle-avoiding paths (Chapter 5). The geometry of static obstacles should be stored in the INSM model, and geometric networks can be generated to avoid obstacles for given users.

- Supports routing on both the logical and the geometric network, which can generate geometric paths based on user-specific logical paths, or re-compute logical paths when geometric paths are inaccessible (Chapter 7). This routing is an addition to the common routing methods such as the shortest path computation.

Two-level routing has shown its ability to flexibly provide accessible paths on user demands. As addressed before, a single complete network cannot provide flexible routing for different types of user. Compared to hierarchical graph methods [HD04, JM05, LOS06b, SSO08, YCDN07, RWS11], my approach clearly defines the criteria on the abstract level (i.e., the logical network), which cannot be skipped in implementation.

There are five topics which need to be discussed regarding the approach and the current implementation, which should be considered in further developments: 1) to introduce other routing criteria; 2) to visualize realistic paths if necessary; 3) to consider subdivision to get spaces of similar size; 4) to consider indoor moving obstacles; and 5) to include real-time applications. The details are listed as follows:

First, more criteria can be introduced such as awareness and orientation ability influencing the selection of geometric paths. In this thesis I developed six criteria (Section 4.3) for routing on the logical network, but more can be designed to include other cases. Regarding geometric paths, a path including fewer changes to orientation is
easy to be followed by a user, thus sensing the abilities of users’ needs to be parameterized for indoor routing. Sensing abilities is related to how a user leverages her/his visual and hearing abilities to achieve indoor wayfinding, which is an indicator to find the easy-to-follow geometric paths. It is necessary to develop related routing criteria to reflect such preferences.

Second, this thesis denotes geometric paths by the straight-lined representation, since a user can perceive a turn (or a corner) as a remarkable ‘landmark’ and follow the paths by directions. In other words, I did not generate a 100% precise trail for a user (see Figure 8.2). Although the precise path can retain every detail as a collision-free path, a user can only rely on an accurate tracking device to follow a precise path without deviation. Considering that the current indoor positioning solutions cannot support very high accuracy, a user cannot localize herself/himself accurately in spaces in real-time to follow the precise path. Thus I decided to compute a simplified but informative path, which provides clear directions/turns and then helps users to understand and follow the paths. If a strict precise path is requested by a user, I can further compute the realistic visualization of the path.

![Figure 8.2](image)

**FIGURE 8.2** A comparison of the schematic and realistic representation of a geometric path. (a) Schematic representation; and (b) Realistic representation. The circles represent the trail of a simplified user.

Third, two-level routing is quite related to the subdivision of a building. If a subdivision results in a few large spaces, then routing in the logical network would provide less information and routing in the geometric network can supplement more details on the computed path. In contrast, when many small spaces are derived from a subdivision, routing in the logical network can already outline the path with these spaces. For example, the shapes of logical paths in the OTB building (with many small spaces) are similar to those of geometric paths (see Chapter 7), and a geometric path is not necessary for a user since the user can easily find her/his way along the logical path. However, logical paths at Schiphol Airport consist of several large spaces where users still need detailed paths inside these large spaces. In general, the granularity of subdivision determines the importance of the logical and geometric network for the routing. For cases of large spaces, a solution is to set a maximum size for single spaces, and then subdivide these large spaces until their sizes are lower than/equal to the maximum size (e.g., meshing).

Fourth, two-level routing can be extended to the computation with indoor moving obstacles. Routing in the geometric network derives accessible and obstacle-avoiding paths for a user. When the boundary or location of indoor obstacles (e.g., a crowd of people or moved furniture) changes, one can locate the changed spaces in the logical network. Then at the current moment a new routing can be conducted in the updated geometric network. As only the changed spaces are updated instead of all the spaces,
the two-level routing approach can promptly adjust and then generate paths according to indoor changes. In the case of continuously moving obstacles, it may not be economic to create new networks all the time. Then a possible solution is to introduce time variable and a prediction model of these moving obstacles: 1) in a period of time, to compute the maximum extent of these obstacle movements with the consideration of a safety buffer; 2) to create a navigation network for them during this period; 3) to store this network with time information. In this way, during a certain period a user can request the computed path based on the stored network.

Fifth, though I only applied two-level routing to normal situations of buildings, the approach is extensible for real-time applications such as an emergency response. In such cases, indoor dynamics need to be considered for building subdivision. This thesis adopts the structural subdivision such as provided by the IFC and CityGML models. However, INSM can also support functional subdivisions, which gives flexibility to defining indoor spaces according to the dynamics. Different conceptual and functional spaces can both be taken into account. I present an example to conceptually discuss the dynamic subdivisions of a building. Figure 8.3 illustrates a serial of subdivisions caused by indoor dynamics. At the initial state all the spaces are accessible to a user. Supposing two sites catch fire, then a new subdivision of the building is performed. The spaces near heat sources are denoted in red. As smoke diffuses along with the fire, another subdivision is made to present the limited safe spaces. As Figure 8.3 shows, the green part has shrunk and more and more red spaces imperilled by fire and smoke. In this example all the three subdivisions follow functional constraints. In such cases, log-
ical and geometric networks can be updated on demand based on safe spaces and new obstacles (e.g., waste) generated from the changes.

§ 8.3 Directions for future research

This section will introduce future work to this PhD research. The future work can be categorized into five aspects:

1. To develop more routing options for two-level routing;
2. To extend the two-level routing approach with new topological relationships, new levels, other semantics, dynamic obstacles and emergency scenarios;
3. To link the two-level routing approach to related research such as indoor guidance and building subdivision;
4. To calibrate the two-level routing results and to pinpoint the application scope;
5. To evaluate the feasibility of two-level routing for new scenarios.

More two-level routing options

This thesis proposes seven options for two-level routing (Section 3.4.3). They are devised by an observation of human behaviours and needs in routing. More options can be investigated and designed for other scenarios by collecting questionnaires from users in different types of building. In these new routing options, two-level routing also needs to flexibly compute paths for different users according to their requirements and preferences.

An extension of the two-level approach is multi-level routing. Theoretically arbitrary levels of indoor space hierarchy can be constructed, which depends on building configuration complexity. More complex buildings may contain more levels. The key for such routing problems is the interaction among all the designed levels.

Extension of the two-level routing approach

Two-level routing relies on two types of navigation networks—the logical and the geometric networks. For logical networks, more topological relationships can be considered for indoor routing, such as adjacency of spaces. When two spaces are not connected but adjacent, the fact indicates the two spaces are close. In addition, adjacency reveals some features of a space: if a space involves many adjacency relationships, which indicates it is ‘contained’ in the building; otherwise, it may be adjacent to outdoors. Thus, the adjacency can be added to new routing criteria on the logical network. For geometric networks, new approaches to give more options can be adopted in more cases. Currently all indoor spaces are represented by 2D polygons. In the future it is necessary to investigate how to apply two-level routing to 3D indoor spaces. For instance, the geometric path should be able to reflect the vertical motion of a human (to get over a chair or get under a desk), or to be able to navigate a drone. In short, two-level routing will be extended by introducing new routing criteria for logical networks,
new creation methods for geometric networks and new routing algorithms for these geometric networks.

Moreover, new levels of details can be introduced to extend two-level routing into multiple-level routing. For instance, an intermediate network representing the connectivity of doors and spaces can be designed in between the logical network and the geometric network, i.e., opening-to-space type. As an opening is a kind of ‘space’, I can put them together with indoor spaces (NUs) to construct a navigation network. Such a network directly represents the relationships among NUs and openings (mostly doors). In this case, a new routing approach should be developed for the three navigation networks to conduct routing on user demands.

In the two-level routing approach, other semantics in addition to the INS’s can be introduced for the logical network and new routing criteria can be developed for these semantics. Semantics of indoor spaces represent different functionalities in distinct data models for specific contexts (e.g., 3DBO and IndoorGML). These semantics can be used to design routing criteria and extend the application scope of two-level routing. For example, a logical network can be assigned with two sets of semantics. Thus, routing on the logical network can be applied to more routing cases.

In addition, two-level routing can be extended to cope with path computation with dynamic (moving and/or unstable–shaped) obstacles. Dynamic obstacles refers to the varying extent of these obstacles, which changes the accessible region of indoor spaces. Thus, routing on both levels needs to follow the real-time boundaries of dynamic obstacles. Spaces heavily influenced by dynamic obstacles should be reflected in the logical network, which results in suitable logical paths for a user; routing in the geometric network generates obstacle-avoiding and accessible paths for the user at a given moment. Two-level routing may need frequent re-computation along with time lapse. Research is needed for synchronization of routing between the two levels with dynamic obstacles. As mentioned before, a simulation model of indoor obstacle dynamics can be introduced. In this case, the varied shape and movement of an obstacle can be predicted along with time lapse. Then a logical and geometric path can be computed at a user’s expected time and locations.

Furthermore, applications in emergency scenarios would be investigated, where both more and fewer openings (due to collapse) are possible. In the logical network, the connectivity of spaces would be impacted by emergencies, and then an updating mechanism is required to maintain nodes and edges of the logical network. As the number of dynamic and static obstacles can sharply increase in a short time, geometric networks should be frequently recreated. In this case, the temporal factor can be introduced to computing geometric paths. For example, a logical network and a geometric network are built for the present, and a new logical and a new geometric network are created after five minutes. In this way, emergency and indoor dynamics can be incorporated in the two-level routing approach by updating logical and geometric networks.

Links with other related research

The two-level routing approach is relevant to other related studies on guidance and building subdivision, though this thesis focuses only on the indoor routing component of the indoor navigation process.
Two-level routing can provide input information for indoor guidance for pedestrians. The common method of guidance for pedestrians is textual directions (e.g., ‘walk forward 100m, then turn left’). Based on a logical path, an abstract route regarding spaces can be organized in landmark-oriented sentences, such as ‘cross Corridor 2, then enter into Office 201’; while for textual directories a geometric path can present more details of the path such as distance and turns. Future research could investigate the combined textual directories generated from the two types of path. For example, only space names are requested by a user due to the simple shapes of the spaces, while both space names and step-by-step instructions are needed for the user in complicated spaces.

Building subdivision can be further studied to support two-level routing. The real-time results of building subdivision can react to two-level routing. A heat map of crowds could be generated when video surveillance is available for a building. On the heat map one can identify hot spots in the building and re-divide new spaces and obstacles (e.g., crowds), which would dynamically derive space subdivision results [ZLS+14, KZ14]. These results influence the logical and geometric network in real-time. Further research can be conducted on the dynamic subdivision automation, and the cooperation between the real-time subdivision and two-level routing.

In addition, it would be interesting to investigate the relationships between indoor and outdoor navigation with the two-level routing approach. Outdoor road networks are naturally geometric networks with different modes for distinct agents (vehicles and pedestrians). It is necessary to investigate and categorize semantics of regions in outdoor environments (e.g., blocks), which may facilitate the generation of logical networks for the outdoors. Furthermore, if two-level routing can be applied to the outdoors, researchers need to design user-related routing options where path computation on the two levels should be defined.

**Correction of the two-level routing results**

Field tests and usability studies should be able to provide an objective evaluation of the results of two-level routing. I can track and monitor user motions and behaviours during indoor routing or wayfinding (without a device), and compare these user tracks with the paths resulting from two-level routing. I can collect the statistics regarding the fitness of the computed paths and the actual user tracks.

Indoor tracking on pedestrians can provide feedback about the practice of users following the computed path. The real trace of users can be used to improve two-level routing. Indoor tracking can be supported by video surveillance [ZZW+16] or indoor positioning equipment. User traces can be extracted from videos [ZZW+16]. Indoor positioning techniques can also continuously collect user locations and support extracting user traces [Xu14]. The processed traces are field data which help the correction of computed logical and geometric paths. The spaces traversed by a user can be compared with the computed logical path, and the trace of the user can be matched to geometric networks to highlight the differences with the computed geometric path. In addition, a real-time indoor tracking system can help users better interact with the two-level routing system to follow the resulting paths. For example, a user is walking along the computed geometric path but she/he meets some difficulties to distinguish the next step. Thus, the user decides to re-plan her/his route by sending the current location to the two-level routing system.
The two-level routing results can be calibrated and the criteria can be improved according to the indoor tracking results. For example, one can collect statistics on the adopted paths of users in a building. Frequently used paths reflect certain user demands and can be compared with the computed ones for the user. Further studies should focus on what factors derive such differences, and then adopt these factors to update the routing criteria.

Another possible study is to pinpoint the application scope of two-level routing. Research could be developed to investigate the complexity of buildings and apply two-level routing to them. For all the buildings, researchers can compute paths by applying two-level routing and then monitor the computed paths that users would like to follow in each building. If only a few users would like to follow the paths in the buildings with certain complexity, this can indicate that two-level routing may not be suitable for such buildings. In this case, routing results would be analysed to conclude whether two-level routing is appropriate to which level of building complexity.

**Evaluate the feasibility of two-level routing for new scenarios**

Researchers may further extend the two-level routing approach to more applications. Two-level routing derives geometric paths on the basis of a visibility graph. Mortari et al. (2014) provide a comparison of the visibility-based networks with centreline-based networks such as Medial Axis Transformation (MAT) [MZLC14]. The centreline-based navigation networks fit for narrow corridor cases, and the visibility-based (e.g., two-level routing) ones are suitable for larger and open spaces. The future work needs to evaluate whether two-level routing is suitable for other new types of application scenario (e.g., with moving obstacles). Some factors can be introduced for tests in these new scenarios, such as the cost of time and compatibility with indoor positioning results.

Other promising directions include:

1. **Authentic 3D routing.** For example, two-level routing can be performed for flying drones which move in complete 3D environments (e.g., above and on surfaces). In such cases, a 3D indoor environment needs to be subdivided according to certain purposes (e.g., the size of a drone and/or importance of the space). Then routing on the abstract level could compute logical paths in the light of certain criteria (e.g., to minimize the total volume of selected spaces). Consequently, the geometric path can be computed, based on the logical path.

2. **Two-level routing which considers ‘optimal’ paths (e.g., the fastest or shortest paths).** On the abstract level, one can investigate how to compute a logical path which contains the ‘optimal’ geometric path. In other words, indoor spaces for routing can be reduced and the optimality of the geometric path can be ensured as well.

3. **The INSM model can be used to analyse other building scenarios for routing in existing buildings.** As addressed before, the INSM is designed to distinguish the use of indoor semantics and this facilitates the generation of navigation networks of two-level routing. Different existing buildings can be enriched with the INSM semantics and then two-level routing can be applied to them.

4. **Real-time factors such as crowd flows and path capacity can be incorporated into two-level routing, which would facilitate evacuation planning in different buildings.** For instance, crowd flows can be regarded as dynamic obstacles which vary over time. Crowd
behaviour should be taken into account to estimate and simulate crowd flow. In different time slots, two-level routing can be employed to compute accessible paths in real-time. In addition, path capacity (i.e., the number of people that can pass through a space) can be evaluated for each space, and new routing criteria could be proposed with regard to the path capacity.
Bibliography


[IZ09] U. Isikdag and S. Zlatanova. Towards defining a framework for automatic generation of build-


the Digital City Special issue of the International Cartographic Association Commission on Geo-Visualization.


Appendix - Proof of three lemmas in Chapter 5

As mentioned in Subsection 5.2, the final convex hull (FCH) contains all the selected obstacles from the start and target locations (e.g., doors). Here I present FCH-related lemmas and proof. The illustration adopted irregular polygons to represent indoor static obstacles (polygons).

**Lemma 1.** If a polygon contains some static indoor obstacles, then the polygon also contains all the visibility edges of the VG derived with these obstacles.

**Proof.** Supposing one of the visibility edges intersects or lies outside the polygon, then there must be (at least) one obstacle vertex outside of the polygon. Because the polygon contains all the selected obstacles, it naturally contains all the obstacle vertices. Thus, the inference of the outside vertex conflicts with this fact, which implies the polygon contains all the visibility edges of these obstacles.

Conversely, if a polygon contains all the visibility edges of the obstacles inside of it, then the polygon contains all the obstacle vertices. Therefore, these obstacles are included in the polygon as well. Therefore, Lemma 1 is proved.

Certainly, the polygon can be a convex hull (CH) of these obstacles and the start and target.

![FIGURE A.1](image)  The shortest path in the bound of a FCH.
Lemma 2. Given a user size, related grouped obstacles, and the start and target in a space, if a computed FCH does not intersect exterior obstacles, then the shortest path from the start to the target is either inside or in the bound of the FCH.

Proof. The FCH does not intersect exterior obstacles (see selecting obstacle groups with FCH in Subsection 5.2), which indicates there are accessible gaps in between the FCH and exterior obstacles. According to Lemma 1, the FCH contains all the visibility edges of the contained obstacles. As the FCH is computed from all the contained obstacle vertices, segments of the FCH are also visibility edges among these obstacles. The shortest path can be computed on the VG consisting of these visibility edges. Thus, the shortest path is either inside or in the bound of the FCH (Figure A.1) and never outside the FCH. Then Lemma 2 is proved.

According to Lemma 2, the shortest path is either contained in the computed FCH or in the bound of the FCH. However, it is not clear whether the path is equal to the shortest path generated from the VG derived with all obstacles in the same space. The following Lemma 3 proves that the two types of shortest path are the same.

Lemma 3. Given a user size, related grouped obstacles, and the start and target in a space, the real shortest path from the start to target is the shortest path derived in the VG of obstacles in the FCH.

Before the proof, I first introduce two notations. The shortest path computed with the VG of obstacles in the FCH is named $P_1$, and the global shortest path computed with all the obstacles in the space is named $P_2$. $P_2$ is the real shortest path. My purpose is to prove $P_1$ is equal to $P_2$.

Proof. $P_2$ has three possible forms: to be completely outside the FCH, completely inside the FCH (including in the FCH bound) or partially inside the FCH. Firstly, if $P_2$ is completely inside the FCH, then Lemma 3 is self-proved.

Secondly, suppose $P_2$ is completely outside of the FCH, which is shown in Figure A.2. Figure A.2 presents a FCH between the start and destination, and $P_2$ is an exterior path to the FCH. I add several auxiliary lines between the FCH and $P_2$, namely, $M_1 N_1, M_2 N_2, \ldots, M_t N_t (t \geq 0)$. Because the FCH is a convex polygon, the auxiliary lines of the FCH’s edges would never enter into the FCH’s interior.

Since the shortest way between two points is a straight line, in Figure 7 I readily have

$$SM_1 + M_1 N_1 < S \ldots N_1$$

where $S \ldots N_1$ denotes the length of all the segments from $S$ to $N_1$ along $P_2$. Analogously, I have

$$M_1 M_2 + M_2 N_2 < M_1 N_1 + N_1 \ldots N_2$$

$$M_2 M_t + M_t N_t < M_2 N_2 + N_2 \ldots N_t$$
$M_t D < M_t N_t + N_t D \quad (8.4)$

**FIGURE A.2** The assumption that $P2$ lies completely outside of the FCH.

Combining the inequalities (1), (2), (3) and (4), a new inequality is as follows:

$$SM_1 + M_1 M_2 + M_2 M_t + M_t D < S...N_1 + N_1...N_2 + N_2...N_t + N_t D \quad (8.5)$$

Equally, there is,

$$SM_1 + M_1 M_2 + M_2 M_t + M_t D < length(P2) \quad (8.6)$$

Inequality (6) indicates that the path from $S$ to $D$ along the FCH (denoted by $SD$) is shorter than $P2$. As I denote the shortest path inside of the FCH (or in the FCH bound) as $P1$, thus there is:

$$Length(P1) \leq SD < length(P2) \quad (8.7)$$

If $P1$ is contained in the FCH, then $P1$ is shorter than the path $SD$ along the FCH; if the 'shortest' path inside the FCH is longer than $SD$, then $P1$ is just $SD$ (Figure A.1). In either case $P2$ is always longer than $P1$, which contradicts the condition ‘$P2$ is the real shortest path’. Therefore, $P2$ cannot be completely outside of the FCH.

Thirdly, suppose $P2$ lies partially inside of the FCH. In Figure A.3, without loss of generality, suppose $P2$ is equal to

$$S...M_1 + M_1...M_2 + M_2...D$$

, which is represented by the dashed lines. I denote the exterior part of $P2$ by $M_1...M_2$ (dashed lines) and the straight line between $M_1$ and $M_2$ by $M_1 M_2$. Apparently, $M_1 M_2 < M_1...M_2$. Thus a path in the FCH, which is denoted by

$$S...M_1, M_1 M_2$$
and $M_2...D$, is shorter than $P_2$. Again, this result contradicts ‘$P_2$ is the real shortest path’. Therefore, $P_2$ cannot be partially inside of the FCH.

Finally, I can conclude the only possibility is $P_2$ lies inside the FCH or in the FCH bound. Then Lemma 3 is proved.

FIGURE A.3  P2 lies partially inside of the FCH. P2 is denoted by dashed lines.
Summary

Humans perform many activities indoors and they show a growing need for indoor navigation, especially in unfamiliar buildings such as airports, museums and hospitals. Complexity of such buildings poses many challenges for building managers and visitors. Indoor navigation services play an important role in supporting these indoor activities. Indoor navigation covers extensive topics such as: 1) indoor positioning and localization; 2) indoor space representation for navigation model generation; 3) indoor routing computation; 4) human wayfinding behaviours; and 5) indoor guidance (e.g., textual directories). So far, a large number of studies of pedestrian indoor navigation have presented diverse navigation models and routing algorithms/methods. However, the major challenge is rarely referred to: how to represent the complex indoor environment for pedestrians and conduct routing according to the different roles and sizes of users. Such complex buildings contain irregular shapes, large open spaces, complicated obstacles and different types of passages. A navigation model can be very complicated if the indoors are accurately represented. Although most research demonstrates feasible indoor navigation models and related routing methods in regular buildings, the focus is still on a general navigation model for pedestrians who are simplified as circles. In fact, pedestrians represent different sizes, motion abilities and preferences (e.g., described in user profiles), which should be reflected in navigation models and be considered for indoor routing (e.g., relevant Spaces of Interest and Points of Interest).

In order to address this challenge, this thesis proposes an innovative indoor modelling and routing approach – two-level routing. It specially targets the case of routing in complex buildings for distinct users. The conceptual (first) level uses general free indoor spaces: this is represented by the logical network whose nodes represent the spaces and edges stand for their connectivity; the detailed (second) level focuses on transition spaces such as openings and Spaces of Interest (SOI), and geometric networks are generated regarding these spaces. Nodes of a geometric network refers to locations of doors, windows and subspaces (SOIs) inside of the larger spaces; and the edges represent detailed paths among these geometric nodes. A combination of the two levels can represent complex buildings in specified spaces, which avoids maintaining a large-scale complete network. User preferences on ordered SOIs are considered in routing on the logical network, and preferences on ordered Points of Interest (POI) are adopted in routing on geometric networks. In a geometric network, accessible obstacle-avoiding paths can be computed for users with different sizes.

To facilitate automatic generation of the two types of network in any building, a new data model named Indoor Navigation Space Model (INSM) is proposed to store connectivity, semantics and geometry of indoor spaces for buildings. Abundant semantics of building components are designed in INSM based on navigational functionalities, such as VerticalUnit(VU) and HorizontalConnector(HC) as vertical and horizontal passages for pedestrians. The INSM supports different subdivision ways of a building in which indoor spaces can be assigned proper semantics.

A logical and geometric network can be automatically derived from INSM, and they can be used individually or together for indoor routing. Thus, different routing options are designed. Paths can be provided by using either the logical network when some users...
are satisfied with a rough description of the path (e.g., the name of spaces), or a geometric path is automatically computed for a user who needs only a detailed path which shows how obstacles can be avoided. The two-level routing approach integrates both logical and geometric networks to obtain paths, when a user provides her/his preferences on SOIs and POIs. For example, routing results for the logical network can exclude unrelated spaces and then derive geometric paths more efficiently. In this thesis, two options are proposed for routing just on the logical network, three options are proposed for routing just on the geometric networks, and seven options for two-level routing.

On the logical network, six routing criteria are proposed and three human wayfinding strategies are adopted to simulate human indoor behaviours. According to a specific criterion, space semantics of logical nodes is utilized to assign different weights to logical nodes and edges. Therefore, routing on the logical network can be accomplished by applying the Dijkstra algorithm. If multiple criteria are adopted, an order of criteria is applied for routing according to a specific user. In this way, logical paths can be computed as a sequence of indoor spaces with clear semantics.

On geometric networks, this thesis proposes a new routing method to provide detailed paths avoiding indoor obstacles with respect to pedestrian sizes. This method allows geometric networks to be derived for individual users with different sizes for any specified spaces.

To demonstrate the use of the two types of network, this thesis tests routing on one level (the logical or the geometric network). Four case studies about the logical network are presented in both simple and complex buildings. In the simple building, no multiple paths lie between spaces A and B, but in the complex buildings, multiple logical paths exist and the candidate paths can be reduced by applying these routing criteria in an order for a user. The relationships of these criteria to user profiles are assumed in this thesis.

The proposed geometric routing regarding user sizes is tested with three case studies: 1) routing for pedestrians with two distinct sizes in one space; 2) routing for pedestrians with changed sizes in one space; and 3) a larger geometric network formed by the ones in a given sequence of spaces. The first case shows that a small increase of user size can largely change the accessible path; the second case shows different path segments for distinct sizes can be combined into one geometric path; the third case demonstrates a geometric network can be created ‘on the fly’ for any specified spaces of a building. Therefore, the generation and routing of geometric networks are very flexible and fit to given users.

To demonstrate the proposed two-level routing approach, this thesis designs five cases. The five cases are distinguished according to the method of model creation (pre-computed or ‘on-the-fly’) and model storage (on the client or server). Two of them are realized in this thesis: 1) Case 1 just in the client pre-computes the logical network and derives geometric networks ‘on the fly’; 2) Case 2 just in the client pre-computes and stores the logical and geometric networks for certain user sizes. Case 1 is implemented in a desktop application for building managers, and Case 2 is realized as a mobile mock-up for mobile users without an internet connection.
As this thesis shows, two-level routing is powerful enough to effectively provide indicative logical paths and/or comprehensive geometric paths, according to different user requirements on path details. In the desktop application, three of the proposed routing options for two-level routing are tested for the simple OTB building and the complex Schiphol Airport building. These use cases demonstrate that the two-level routing approach includes the following merits:

- It supports routing in different abstraction forms of a building. The INSM model can describe different subdivision results of a building, and it allows two types of routing network to be derived – pure logical and geometric ones. The logical network contains the topology and semantics of indoor spaces, and the geometric network provides accurate geometry for paths. A consistent navigation model is formed with the two networks, i.e., the conceptual and detailed levels.

- On the conceptual level, it supports routing on a logical network and assists the derivation of a conceptual path (i.e., logical path) for a user in terms of space sequence. Routing criteria are designed based on the INSM semantics of spaces, which can generate logical paths similar to human wayfinding results such as minimizing VerticalUnit or HorizontalConnector.

- On the detailed level, it considers the size of users and results in obstacle-avoiding paths. By using this approach, geometric networks can be generated to avoid obstacles for the given users and accessible paths are flexibly provided for user demands. This approach can process changes of user size more efficiently, in contrast to routing on a complete geometric network.

- It supports routing on both the logical and the geometric networks, which can generate geometric paths based on user-specific logical paths, or re-compute logical paths when geometric paths are inaccessible. This computation method is very useful for complex buildings. The two-level routing approach can flexibly provide logical and geometric paths according to user preferences and sizes, and can adjust the generated paths in limited time.

Based on the two-level routing approach, this thesis also provides a vision on possible cooperation with other methods. A potential direction is to design more routing options according to other indoor scenarios and user preferences. Extensions of the two-level routing approach, such as other types of semantics, multi-level networks and dynamic obstacles, will make it possible to deal with other routing cases. Last but not least, it is also promising to explore its relationships with indoor guidance, different building subdivisions and outdoor navigation.
Samenvatting


Om deze uitdaging aan te gaan wordt in deze dissertatie een innovatieve aanpak van indoormodelering en routeplanning voorgesteld, te weten routeplanning op twee niveaus. Het richt zich met name op de routeplanning in complexe gebouwen ten behoeve van verschillende gebruikers. Het conceptuele (eerste) niveau maakt gebruik van algemene vrije indooromruimten: het wordt geregistreerd door het logisch netwerk waarvan de nodes (knooppunten) de ruimten representeren en de edges (verbindingen) hun connectiviteit aangeven. Het gedetailleerde (tweede) niveau richt zich op transitieruimten, zoals doorgangen en Space of Interest (SOI). Geometrische netwerken worden gegenereerd voor deze verzameling van ruimten. Nodes van een geometrisch netwerk verwijzen naar de plaats waar zich deuren, ramen en subruimten (SOIs) in de grotere ruimten bevinden. De edges representeren gedetailleerde verbindingen langs deze geometrische nodes. Een combinatie van de twee niveaus kan complexe gebouwen representeren waardoor het onderhouden van een omvangrijk netwerk kan worden vermeden. Gebruikersvoorkeuren voor aangegraagde SOIs worden in de routeplanning van het logisch netwerk meegegenomen. Voorkeuren voor aangegraagde Points of Interest (POIs) in routeplanning worden in geometrische netwerken opgenomen. In een geometrisch netwerk kunnen toegankelijke obstakel-vermijdende verbindingen voor gebruikers van verschillende omvang worden berekend.

Om de automatische generatie van de twee netwerktypes in welk gebouw dan ook te faciliteren wordt een nieuw datamodel - het Indoor Navigation Space Model (INSM) - voorgesteld om connectiviteit, semantiek en geometrie van indoorruimten voor gebouwen
op te slaan. Volledige semantiek van gebouwcomponenten wordt ontworpen in het INSM gebaseerd op navigeerbare eenheden, zoals VerticalUnit(VU) en HorizontalConnector(HC) als verticale en horizontale doorgangen voor voetgangers. Het INSM ondersteunt verschillende opdelingen van een gebouw waarbij indoorruimten aan de juiste semantiek worden gekoppeld.

Een logisch en geometrisch netwerk kan automatisch worden afgeleid van het INSM. Deze kunnen onafhankelijk van elkaar of gezamenlijk voor indoorrouteplanning worden gebruikt. Er zijn dus verschillende opties voor routeplanning ontworpen. Als gebruikers genoeg nemen met een ruwe beschrijving van de route (bijv. de naam van ruimten) kunnen de routes worden weergegeven door gebruik te maken van het logisch netwerk. Een geometrische route wordt automatisch berekend voor een gebruiker die een gedetailleerde route, waarin wordt aangegeven hoe obstakels kunnen worden vermeden, nodig heeft. Als een gebruiker zijn/haar voorkeuren voor SOIs en POIs opgeeft integreert de routeplanning op twee niveaus zowel het logisch als het geometrisch netwerk om routes te genereren. Zo kunnen routeplanningsresultaten in het logisch netwerk ruimten die er niet toe doen uitsluiten en vervolgens geometrische routes meer efficiënt afleiden. In deze dissertatie worden voor routeplanning twee opties in alleen het logisch netwerk, drie opties in het geometrisch netwerk en zeven opties op twee niveaus voorgesteld.


Voor geometrische netwerken wordt in dit proefschrift een nieuwe routeplanningsmethode voorgesteld om gedetailleerde routes met indoorobstakels voor voetgangers met een verschillende omvang te identificeren. Deze methode maakt het mogelijk geometrische netwerken voor individuele gebruikers met een verschillende omvang voor elke gepecifieerde ruimte af te leiden.

Om het gebruik van de twee typen netwerken te illustreren wordt in dit proefschrift de routeplanning eerst op één niveau (het logisch of het geometrisch netwerk) getest. Vier casestudies inzake het logisch netwerk voor zowel eenvoudige als complexe gebouwen worden gepresenteerd. In eenvoudige gebouwen liggen geen meervoudige paden omvang tussen ruimte A en B. In complexe gebouwen bestaan meerdere logische routes en de kandidaat routes kunnen worden verminderd door het toepassen van deze routeplanningscriteria in volgorde van prioriteit voor een gebruiker). In deze dissertatie wordt van de relaties van deze criteria met de gebruikersprofielen uitgegaan.

De voorgestelde geometrisch routeplanning met betrekking tot de omvang van de gebruikers is in drie casestudies getest: 1) routeplanning voor twee voetgangers van verschillende omvang in één ruimte; 2) routeplanning voor een voetganger van veranderende omvang in één ruimte; 3) een groter geometrisch netwerk dat gevormd wordt voor koppeling van een opeenvolgende serie van ruimten. De eerste casus toont
aan, dat een kleine toename van de omvang van de gebruikers een grote verandering van de route kan opleveren; de tweede casus heeft verschillende routesegmenten voor onderscheiden omvang waarmee een combinatie in één geometrische route mogelijk wordt gemaakt; de derde casus geeft een geometrisch netwerk dat on the fly voor iedere gespecificeerde ruimte van een gebouw beschikbaar is. Hierdoor worden de generatie en routeplanning van geometrische netwerken zeer flexibel en toepasbaar voor de gebruikers.

Om de voorgestelde routeplanning op twee niveaus te illustreren worden er vijf cases in deze dissertatie geïntroduceerd. Deze vijf cases worden conform de modelcreatie (vooraf berekend of on the fly) en modelopslag (in de client of de server) onderscheiden. Twee hiervan worden in deze dissertatie gerealiseerd: 1) In casus 1 wordt in de client alleen het logisch netwerk vooraf berekend en worden geometrische netwerken on the fly afgeleid; 2) In casus 2 worden alleen in de client de routes vooraf berekend en opgeslagen in het logisch en geometrisch netwerk voor gebruikers van een bepaalde omvang. Casus 1 is op een desktopapplicatie voor gebouwbeheerders geïmplementeerd en casus 2 is als een mobiele mock-up voor mobiele gebruikers zonder internetverbinding gerealiseerd.

Zoals in deze dissertatie aangetoond is de routeplanning op twee niveaus krachtig genoeg om ruwe logische routes en/of gedetailleerde geometrische routes conform de verschillende gebruikerseisen effectief aan te bieden. In de desktopapplicatie zijn drie van de voorgestelde opties voor routeplanning op twee niveaus getest voor het eenvoudige gebouw van het OTB en het complexe gebouw van de luchthaven Schiphol. Deze gebruikscases tonen aan dat de aanpak van de routeplanning op twee niveaus de volgende voordelen oplevert:


- Op het conceptuele niveau wordt routeplanning in een logisch netwerk ondersteund. Het assisteert het afleiden van een conceptueel route (i.c. logische route) voor een gebruiker als serie opeenvolgende ruimten. Gebaseerd op de INSM semantiek van ruimten, waarmee logische ruimten gelijk aan menselijke oriëntatie- en navigatiereultaten genereerd kunnen worden (zoals het minimaliseren van gebruik VerticalUnit of HorizontalConnector) zijn routeplanningscriteria ontworpen.

- Op het gedetailleerde niveau wordt de omvang van gebruikers meegenomen, dat resulteert in het vermijden van obstakels op de routes. Door van deze aanpak gebruik te maken kunnen geometrische netwerken worden gegeneerd om obstakels voor de gebruikers te omzeilen. Toegankelijke routes worden flexibel op gebruikersverzoek aangegeven. Deze benadering kan de wijzigingen van de omvang van gebruikers efficiënter verwerken. Dit in tegenstelling tot routeplanning via een compleet geometrisch netwerk.

- Het ondersteunt routeplanning zowel in het logisch als in geometrisch netwerk. Hiermee kunnen geometrische routes gebaseerd op gebruikers specifieke logische routes
worden gegenereerd of kunnen logische routes als geometrische routes niet beschikbaar zijn opnieuw worden berekend. Deze berekeningswijze is erg nuttig voor complexe gebouwen. De routeplanning op twee niveaus heeft de mogelijkheid om flexibel logische en geometrische paden conform gebruikersvoorkeuren en omvang aan te bieden en kan de gegenereerde paden in korte tijd aanpassen.

Op basis van de routeplanning op twee niveaus wordt in deze dissertatie ook een visie gegeven voor mogelijke integratie met andere methoden. Een potentiële richting is het ontwerpen van meer routingplanningsopties conform andere indoorscenarios en gebruikersvoorkeuren. Uitbreiding van de routeplanning op twee niveaus, zoals andere semantiektypen, meerlaagse netwerken en dynamische obstakels, zal het mogelijk maken om met andere routeplanningscases te integreren. Tenslotte is het zeker de moeite de waarde van de relaties met indoorbegeleiding, verschillende opdelingen van gebouwen en outdoor navigatie verder te onderzoeken.
Curriculum Vitae

Liu Liu was born in Kaiyang, China, on January 11, 1986. He graduated with a BSc in Surveying Engineering at the Tongji University in Shanghai in 2007. He then did his MSc in Geographic Information System at the Beijing Normal University in Beijing, and graduated in 2010. Afterwards Liu joined the GIS section of the OTB Research Institute in Delft and started his PhD project Semantic Modelling for Indoor Routing, funded by the China Scholarship Council (CSC), and supervised by Peter Van Oosterom and Sisi Zlatanova. In October 2014, Liu visited Bentley System Inc. headquartered in Exton, USA, and collaborated with Mark Anderson and Mark Smith on an indoor navigation project through 2015. The work of this project later became a part of Chapter 7 of this thesis.

Liu enjoys working on indoor navigation studies and spatial analysis problems. Now he is continuing the research of modelling, navigation and spatial analysis of indoor environments.