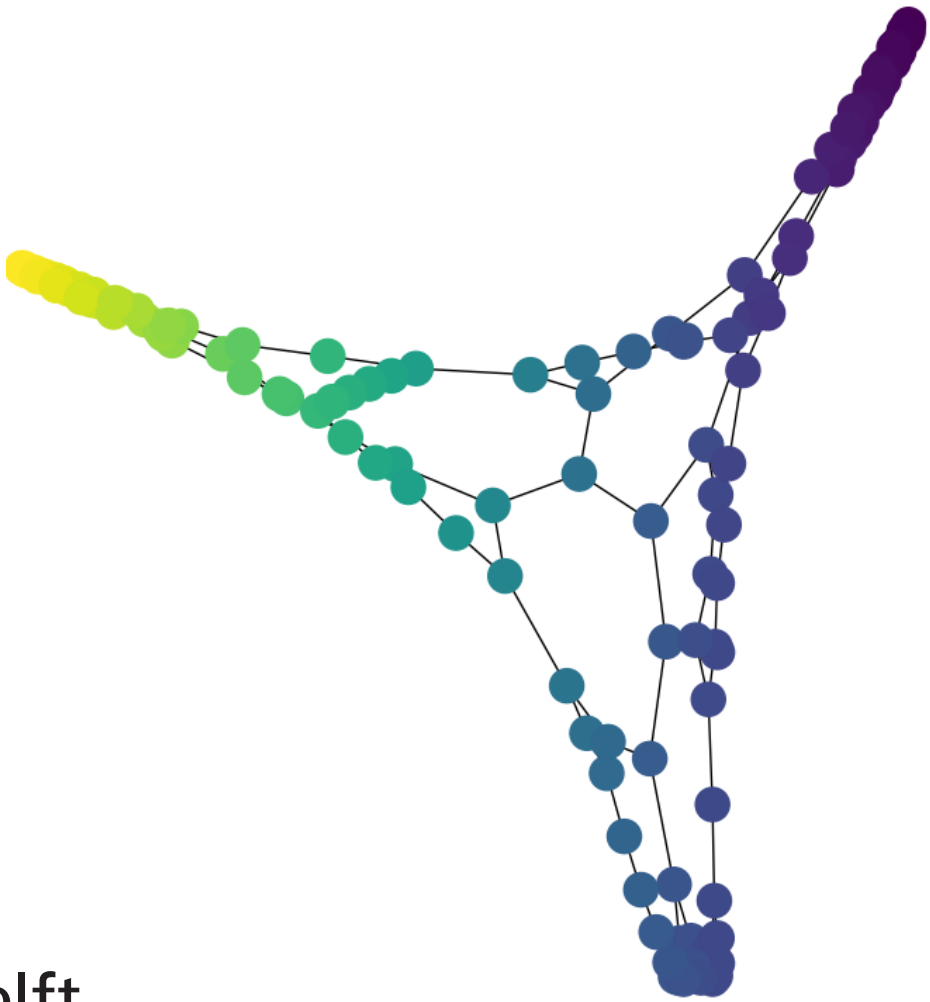
Master of Science in Geomatics for the Built Environment

A GRAPH-MATCHING APPROACH TO INDOOR LOCALIZATION

USING A MOBILE DEVICE AND A REFERENCE BIM

Francina Johanna Bot



TU Delft

A GRAPH-MATCHING APPROACH TO INDOOR LOCALIZATION

USING A MOBILE DEVICE AND A REFERENCE BIM

Graduation Thesis

Master of Science in Geomatics for the Built Environment
Delft University of Technology

by

Francina Johanna Bot

March 7, 2019

Fanny Bot (2019): *A graph-matching approach to indoor localization: using a mobile device and a reference BIM*

Supervisors:	E. Verbree	MSc	GIS Technology
	P. Nourian	PhD	Design Informatics
	A. A. Diakité	PhD	3D Geoinformation Group
Co-reader:	R. Y. Peters	PhD	3D Geoinformation Group

Abstract

Indoor localization provides for a much researched subject, as the complexity and size of many public buildings require extensive and properly designed methods to facilitate location specific processes. Indoor localization entails finding a qualitative description of the occupied area. One that is human interpretable, rather than a quantitative position in Euclidean space. In other words, the context of an indoor environment has to be understood, such that a position can be transcended to a meaningful location.

A space is defined as a mathematical structure with relational properties, to which all its members adhere. As a subset of space in general, topological space is a structure describing the relationships between (parts of) objects that do not change under continuous transformation. This also defines metric space, as a set with a global distance function, where distances and angles between all of its elements are defined. Indoor space can than be interpreted as a structure bounded by physical or functional elements, enabling human activities. It should entail the geometric place an actor is in, the topological structure that place is a part of, and the semantics giving the place meaning.

Many indoor positioning methods have been developed, which can provide an actor with a relative geometric place. Most preferred are positioning systems not relying on a contingent system, which can be performed using a hybrid fusion of sensors embedded into a mobile device. Such a system found to perform sufficiently is VI-SLAM, simultaneously building a geometric place and tracking each pose and heading relatively. Its output can be formed as a mesh model, in which a viewshed of the indoor environment is built.

From a mesh model, a topological structure can be derived in the form of its dual graph. Now to finalize this representation of indoor space which can be captured using a mobile device, it has to be enriched with a meaningful context. These semantics are generally stored in BIM models, as a building representation based on AEC best practices. Thus to transcend the position retrieved using SLAM to a location, is has to be matched with a BIM model, so that the appertaining semantics can be connected.

The method proposed in this research provides for a possibility to perform graph-based indoor localization, by extracting a graph from both input sources and comparing them, in order to find a match. However different in nature and structure, both input sources can be converted to a graph of similar calibre, such that they can be tested for a match. All operations performed on the graphs are derived from spectral graph theory. The graph simplification and analysis is performing using the eigen spectrum of the graph Laplacian, and the match is performed by remapping the spectral graphs into a

vector sub space using the eigen spectrum of the data covariance matrix.

After a match between both graphs is found, the current position of the actor within the mesh model can be translated to the room found in the graph. This room is now connected to a room within the reference graph, for which the semantics are stored in the BIM. Returning these to the actor, a location description is formed.

Acknowledgements

The completion of this thesis has been an extensive process, which I could not have completed without the support of the people around me.

First, I would like to thank my supervisors for their advice in my research, and their understanding when I had to endure some setbacks. Specifically, thanks to Edward Verbree for his help in maintaining the course and always reminding me what the final goal of this research was. Thanks to Pirouz Nourian for the intensive learning sessions, these greatly helped me in achieving my goals. Thanks to Abdoulaye Diakité for getting me started on the research and exciting to dig into the topic. Thanks to Ravi Peters for his helpful remarks at the end of the research. And thanks to Stefan van der Spek for stepping in when we needed a delegate at friday late afternoon.

Furthermore, I would like to express my gratitude to the student counsellors Sietske Sibie and Stephanie Meulemans, who helped me along practically and organisationally during the extent of my graduation process. Also, thanks to fellow students Birgit Ligtvoet, Bart Staats and Rob Braggaar for the study sessions we held. You inspired and motivated me greatly. And finally I would like to thank my fiancé Rafaël Straayer for the mental support and encouragement.

Contents

List of Figures

List of Tables

Acronyms

6DoF	Six Degrees of Freedom I
AAL	Ambient Assistent Living 12
ADF	Area Description File 58
AEC	Architecture, Engineering & Construction 18
API	Application Programming Interface I
AR	Augmented Reality 2
ASCII	American Standard Code for Information Interchange 63
BIM	Building Information Model 4
BK	the faculty of Architecture & the Built Environment 56
COM	Concurrent Odometry and Mapping 3
EWI	the faculty of Electrical Engineering, Math and Computer Sciences 56
FIG	General Assembly of the International Federation of Surveyors 11
FOV	Field of View 29
GIS	Geographic Information System 11
GNSS	Global Navigation Satellite System 24
GPS	Global Positioning System
GSM	Global System for Mobile Communications 24
LBS	Location Based Services 3
IFC	Industry Foundation Classes 23
IMM	Indoor Mapping and Modelling 2
IMU	Inertial Measurement Unit 10
IR	Infra Red 26
ISO	International Standardisation Organisation
LBS	Location Based Service 3
LTE	wireless communication using Long Term Evolution 24
MEMS	Micro Electro Mechanical Systems 25

OGC	Open Geospatial Consortium
RFID	Radio Frequency Identification 24
RGB	Red-Green-Blue Imaging 26
RGB-D	Red-Green-Blue-Depth 25
SDK	Software Development Kit I
SLAM	Simultaneous Localization and Mapping 3
SPLAM	Simultaneous Planning, Localization and Mapping 26
STEP	STEP-file 63
UMTS	Universal Mobile Telecommunication System 24
VI-SLAM	Visual Inertial Simultaneous Localization and Mapping 6
VIO	Visual-Inertial Odometry 27
WLAN	Wireless Local Area Network 24

1 Introduction

The release of Tango technology by Google provoked the interest of many, as it provided opportunities for making indoor navigation widely attainable, as well as easily available to the public. Tango technology was launched in 2014 as a platform for mobile 3D tracking and perception (Google, 2017a).

During Google's 2015 I/O presentation, a lively picture for the necessity of Tango technology was painted by Johnny Lee (Lee, 2015): "One of the core beliefs and ideas behind this project is the fact that, when we place our mobile phones on a table, they do not actually understand the things that are around them. They do not understand the extents of the table, or the room that they are in." From this stance rises the aim of the project: to develop "the hardware and software technologies to help everything and everyone understand precisely where they are."

This sense of space is to be reached "**using computer vision to give devices the ability to understand their position relative to the world around them**" (Google, 2017b). The Lead Product Manager for Tango, Larry Yang, has stated: "We think that computer vision to mobile devices is an inevitability. We're trying to get ahead of that curve a little bit by creating the software development platform that integrates the motion tracking and the depth sensing to be able to create these interesting experiences" (Eddy, 2015).

Furthermore, Tango technology was released accompanied by an open call for developers to create applications that "explore physical space around the user, including precise navigation (FIGURE 1.1) without GPS, windows into virtual 3D worlds, measurement and scanning of spaces, and games that know where they are in the room what's around them" (Google, 2017b).

In order to showcase what Tango technology envisioned for indoor navigation, three museum experiences were created for Tango enabled devices (FIGURE 1.2). The museums lend Tango phones to visitors, with which they could navigate through specifically de-

FIGURE 1.1: Envisioned navigation functionality for Tango Technology. (Google, 2017a)

FIGURE 1.2: Example of an AR museum navigation application in the Museu National d'Art de Catalunya in Barcelona. (Retrieved from http://blog.guidigo.com/blog/guidigo-presents-the-first-project-tango-app-capable-of-3d-indoor-geolocation/).

signed expositions. The applications existed of a limited set of Augmented Reality (AR) overlays of pre-set routes, visualisations and additional information.

Though this is an example of indoor navigation on a commercially available handheld device, the applications used are specifically designed for the exposition in question. The navigation markers were presented as a direct product of exploration, when in fact they are predefined as AR elements connected to a set of 2D landmarks found by image matching. A more comprehensive navigation solution would create an active data connection between what data the visual system can capture and additional stored information about the location. Using Indoor Mapping and Modelling (IMM) best practices the model containing this information can be, at least partially, created using visual systems. However, this process requires human involvement to account for the interpretation of input data. Thus, a navigation model cannot be directly extracted from environment exploration (FIGURE 1.3), as it requires the construction of a context to navigate through. It is this context which transcends a position into a meaningful location, which can provide an actor with sufficient information for navigation.

The Tango project was discontinued in August of 2017, due to commercial reasons, and replaced by the new ARCore (Google, n.d.). With this new project, the focus was reshifted to pure AR functionality on mobile devices (Matney, 2017). The motion tracking

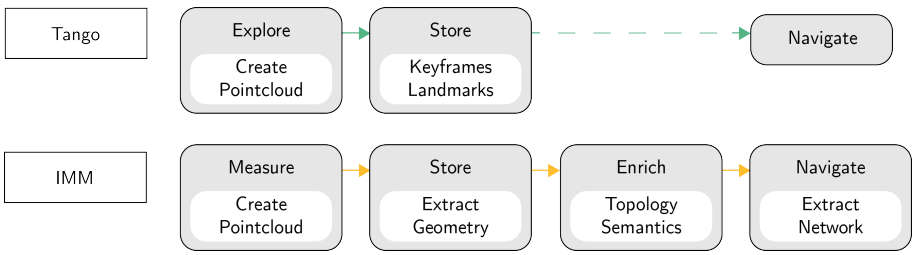


FIGURE 1.3: Misalignment in the conceptual framework of the generation of a navigation model between Tango aims, based on mobile robotics, and IMM common practices.

methods applied were redefined as executing Concurrent Odometry and Mapping (COM) in stead of Simultaneous Localization and Mapping (SLAM), or entailing the exploration part of robot navigation (FIGURE 3.3).

Nonetheless, the notion of reaching an **understanding of space** using technologies available on a mobile device remains interesting. A plethora of Location Based Service (LBS), safety monitoring, surveying and other applications can be imagined, and navigation through a museum exposition would only be the tip of the iceberg. If a digital interpretation of the captured elements can be generated, a certain degree of understanding of space can be reached.

1.1 Problem Definition

Indoor localization is a much researched subject (Xiao, Zhou, Yi, & Ni, 2016; Yassin et al., 2016; Zafari, Gkelias, & Leung, 2017; Zlatanova et al., 2013), as the complexity and size of many public buildings require extensive and properly designed methods to facilitate location specific processes (Lemmens, 2013; Mautz, 2012; Zlatanova, Liu, Sithole, Zhao, & Mortari, 2014). The methods created for such services outdoor cannot be translated directly, new definitions of each component need to be defined (Mautz, 2012). A proper localization process requires positioning into a semantically rich model of the physical environment, or the simultaneous generation of both (Fuentes-Pacheco, Ruiz-Ascencio, & Rendón-Mancha, 2012; Lemmens, 2013). In other words, the context of an indoor environment has to be understood, such that a position can be transcended to a meaningful location.

The target user for such a process would be any actor executing any type of task inside a public building, thus an indoor localization system should be made easily attainable to anyone. As the use of smartphones is widespread, the capabilities of such devices lend themselves for outdoor localization as well, and many LBS are specifically aimed at smartphones, ideally such a system would be designed in order to function on a mobile device.

Thus, a mobile indoor *positioning* system should be applied, along with the availability of a *contextual* 3D building model to position the actor in, so that the combined information defines the actors *location*. Part of the *positioning* system could be expected to function autonomously, though the application of a contingent system greatly increases positioning accuracy (Lemmens, 2013; Mautz, 2012; Yassin et al., 2016). A hybrid could be composed, which would not require the instalment of a new network, and which can function using the sensors typically built into a smartphone. An autonomously operating process deemed sufficient for indoor positioning is found in SLAM, especially when based on the integration of several different sensors (Fuentes-Pacheco et al., 2012; Zlatanova et al., 2013). If an actors' position can be integrated into a *contextual* map of indoor space, the location can be determined. Such a map should be comprised of geometry, topology and semantics (Isikdag et al., 2013). The geometry then represents the physical boundaries of indoor space, while the topology describes the connectedness and nearness of the elements that form these boundaries. Semantics may represent the functional boundaries of indoor space, as well as the context that transcends a position into a location.

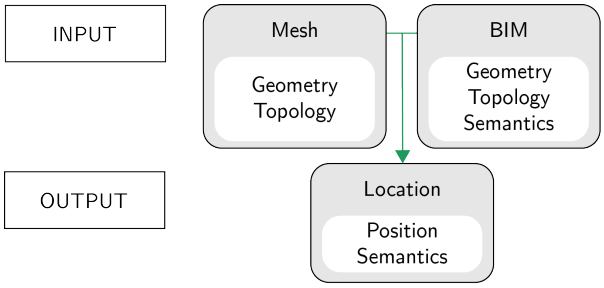


FIGURE 1.4: A conceptual depiction of the general solution proposed to performing indoor localization. By matching the geometries of a polygonal mesh created on-the-fly and the same element in a BIM model, an indoor location can be generated.

An autonomous SLAM process can output a polygonal mesh created on-the-fly as a model of indoor space, containing its geometry and topology. The output model is used as a map in which an actor can be positioned. Semantics describing building geometry and topology are generally stored in a Building Information Model (BIM). The generation of a real-time position of the actor operating the hand-held device into the semantically rich map would generate active indoor localization.

The main objective of the research would then be to align a scan of an indoor environment, created on-the-fly, with a contextually rich model of the building that contains this indoor environment. The actor capturing the scan would then be able to retrieve all semantic information available about the environment itself and the objects it contains. An overview of this connection between the available input, generated to create the desired output can be found in FIGURE 1.4.

1.2 | Research Objectives

The aim of this research is to explore possibilities of performing indoor localization using a mobile device. More specifically, the aim is to develop a method to teach a mobile device to understand its surroundings. This is to be reached by capturing the context and all of its relevant meaning, in order to transcend data to information. As the relevance of information is application dependent (Afyouni, Ray, & Claramunt, 2012), it should be grouped and retrievable by a single marker, i.e.to find a location would be to find the an id connected to information about the room an actor is in. Therefore, an established standard should be used to store contextual information, so that each application can be type-focused and be built upon the same principles. Then, the accessibility of contextual information while on location enables a wide range of possible applications. The main objective on a data level would then be to generate this single marker as a representation of the connection between an indoor environment and an information source describing its context. A method to perform this process is proposed, which is built on best practices and underlying principles, taken from an interdisciplinary viewpoint. Sub-goals are defined in order to create a foundation for this method, and each sub-goal is translated into a sub-question guiding the research.

1.2.1 | Understanding of Space

The subject is inspired by the underlying ambitions of Google's project Tango (PAGE 1), specifically the notion that computer vision on mobile devices can potentially reach an understanding of space. In order to define what this understanding should entail, a theoretical inquiry is taken on the definition of space, its composition, and its context. A conclusion is drawn as to what kind of data can represent indoor space such that it may provide a basis for indoor localization.

1.2.2 | Indoor Positioning using a Mobile Device

The location of an actor is determined by his position, enriched with semantic information. Thus, a mobile device performing indoor localization should be capable of finding its indoor position, and allow for a transcendence of said position to a location using contextual information. Many of the indoor positioning systems that have been implemented thus far are contingent systems, usually depending on a specifically installed infrastructure (Lemmens, 2013; Mautz, 2012; Yassin et al., 2016). These techniques would only become commercially interesting if they can provide the consumer with an affordable and easily attainable solution, as currently owning a smartphone is the only requirement to utilize outdoor positioning techniques (Mautz, 2012; The Economist, 2012). Thus, in order to make indoor localization easily attainable and available to the public, it should be based on a well operating indoor positioning system. The system should be efficient in both processing speed and memory usage, as it should function on a mobile device in real-time. As the application of SLAM in robotics provides a solution basis for indoor positioning (Zlatanova et al., 2013), its functionality is explored theoretically as well as experimentally.

1.2.3 | Transcending a Position to a Location

Indoor localization entails understanding of the environment an actor is situated in, and decisions an actor makes can be advised using this information. Thus, a model used for localization should be able to provide an actor with semantic information about surrounding elements, and to update which information is displayed according to the actors' movement. This research includes a proposal for a way in which such an active connection can be made. The principles behind the connection are based on the definition of indoor space and the functionality of real-time indoor positioning using SLAM, as their culmination provides for a location. Thus, data characteristics that define indoor space should be aligned, in order to match the data itself. The way in which the data can be matched depends on the meaning behind each element, the structure in which it is placed, and what characteristics are necessary in order to generate an appropriate match. Furthermore, a positive outcome of a match should result in a localization output, as the placement of the actor into the digital representation of the indoor environment. This can be done either by returning semantic information to the user, or by registering the captured data of the environment into the reference model, or both.

1.3 Problem Statement

The starting point of this research is the mesh model resulting from performing Visual Inertial Simultaneous Localization and Mapping (VI-SLAM) on a hand held device. As much research is available on analysing geometric structures in order to gain an understanding of the indoor space it represents, the first approach was aimed at matching the geometries of the mesh and reference models.

A first interpretation was to filter out objects that could be translated into BIM elements, as often used in the creation of such models representing an as-built state (Volk et al., 2014). Examples of extracting doors and walls from mesh models or point clouds are Xiong, Adan, Akinci, and Huber (2013) and Thomson and Boehm (2015), where these elements are found and directly translated into the output geometry type. Díaz-Vilariño, Conde, Lagüela, and Lorenzo (2015), Hong et al. (2015), Khoshelham and Díaz-Vilariño (2014) and Zeibak-Shini, Sacks, Ma, and Filin (2016) use the planar and regular composition of many building structures in order to extract walls and floors. Many of these methods begin with semi-automatic registration of input data. However useful while post-processing, an approach where data registration is not automated would not provide for a sufficient basis for indoor localization on a hand-held device, preferably executable on-the-fly.

Díaz-Vilariño, Khoshelham, Martínez-Sánchez, and Arias (2015) use the intrinsic properties of a point cloud to determine its interior envelope, thus finding the direction of the three main axes in the data. Based on these vectors, a rotation matrix can be built in order to align the data to a new set of main axes. Finding these parameters would automate model registration. Then, extracting building elements from the dataset would result in finding geometry that can be matched with the reference model.

Inquiry of the data captured for this research proves that the model is based on gravity-aware scanning. In other words, the normal vectors of the mesh pointing upwards should and do represent the floor (FIGURE 5.6). Furthermore, the general direction of normal vectors anchored on each wall corresponds. However, finding the interior envelope of the data based on clusters in normal directions does not lead to a rotation matrix useful for registration to the world origin. This due to the fact that the model does not entail the full enclosure of the indoor environment it represents. Realistically, neither would any model captured using a hand-held device, due to actor handling and the limited FOV. However, if the model cannot be registered, its geometry cannot be aligned and matched with reference geometry. Perhaps machine learning algorithms used to register large sets of images would be capable of performing this action.

When handling the geometry of mesh models, a large factor is maintaining its topological integrity (Botsch et al., 2010). As the topological structure behind the geometry holds a lot of important properties about the shapes captured, it should be maintained through analysis and deformation. Though mesh handling common in computer graphics, model building in IMM often entails handling pointclouds and even extracting these from mesh models. This semi structured data type has proven sufficient for segmentation and extraction of building elements to build reference models. Furthermore, these data points are easier to handle than the complex structures of mesh models. However, already embedded topological information is again lost when performing these analyses.

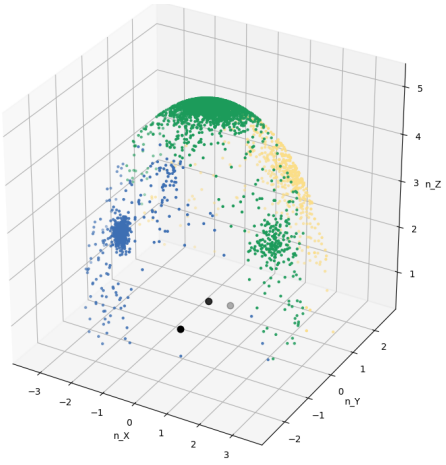


FIGURE 1.5: 3D scatterplot of normal vector directions for the mesh model in FIGURE 5.4. The colours represent the clusters found using k-means, the black dots their cluster centres. The clusters are not distinct enough to use as a basis for a rotation matrix describing the data's exterior envelope.

A manner in which easy to handle data points can be used without losing the intrinsic topological information of a dataset, is by extracting the dual graph of a mesh model. The graph structure itself represents the composition of the model as well as many other intrinsic properties, while geometric placement can be kept as an attribute (FIGURE 5.7). Furthermore, no matter how different representations of an indoor environment may be, the underlying structures should be the same, or at least similar, as the underlying structure is that of the indoor environment itself.

The problem of transcending a position to a location becomes one of comparing the structure of a set of captured positions to the structure of a reference location. From a data perspective, a mesh model captured using a mobile device represents a set of positions, while the added semantics transcends the geometry and topology of a BIM model to a location. Extracting a graph to represent the underlying structure from both models and matching these, would then provide for a means to perform indoor localization.

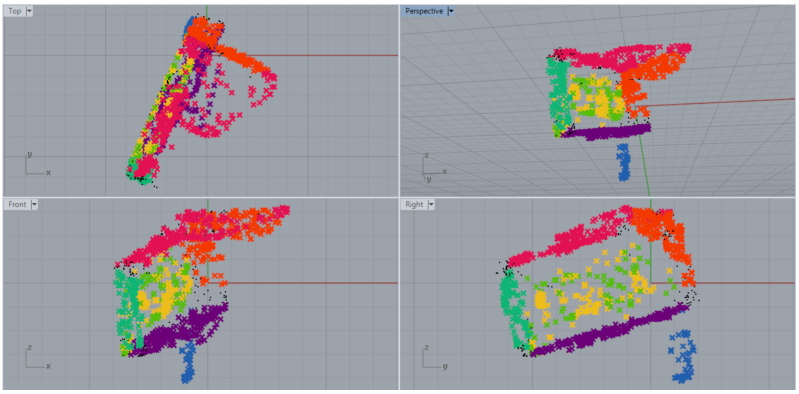


FIGURE 1.6: Segmentation of a point cloud extracted from the mesh model in FIGURE 5.4.

1.4 Research Questions

The main question to be answered is defined as follows:

IN WHAT WAY CAN GRAPH-BASED INDOOR LOCALIZATION BE PERFORMED USING A MOBILE DEVICE, BY COMBINING SLAM AND A BIM MODEL?

The problem is broken down into a theoretical and a practical partition, where theoretical research is executed in order to outline how data can define an indoor location. The localization process is then interpreted as a data-matching problem, connecting the output of real-time SLAM on a mobile device to a reference BIM model. According to the sub-goals, the problem can be broken down into the following sub questions:

- WHAT CHARACTERISTICS OF INDOOR SPACE NEED TO BE UNDERSTOOD IN ORDER TO LOCALIZE AN ACTOR?
- IN WHAT WAY CAN PERFORMING VI-SLAM A MOBILE DEVICE BE UTILIZED AS A BASIS FOR INDOOR LOCALIZATION?
- IN WHAT WAY CAN A GRAPH MODEL OF AN INDOOR ENVIRONMENT BE ENRICHED WITH SEMANTIC INFORMATION FROM A BIM?

1.5 Research Scope

The research focuses at facilitating indoor localization, by defining the process as a data matching problem. From this point of view stem the following assumptions and problem perimeters:

- Localization is interpreted as positioning into a contextually rich map, containing geometry, topology, and semantics (§ 2.2.2).
- The SLAM process is assumed to be capable of performing the positioning, and only its output as a polygonal mesh is taken into account. Improvements on SLAM or VI-SLAM as well as known issues in the process are out of scope.
- The output of the SLAM process contains a current position and trajectory of the user, as points in $\mathbb{R}^3$ set over time. These points are stored relative to each point describing the mesh model. As such, SLAM is assumed to perform a positioning process. The method is devised to transcend this set of positions into a location, so that the current position can be mapped to a location. Thus, finding and tracking a position using SLAM is left out of scope.
- The main source for the contextually rich map is a BIM model, as a data standard increasingly used to represent indoor environments. The model is assumed to represent the as-built state, and to be modelled according to IFC requirements. The use

of any other type of reference model, or a comparison between types of models are out of scope.

- Any drawback to the sensors embedded into the mobile device is left out of scope. Environmental conditions have influenced the quality of captured data and are mentioned the experiment in appendix Appendix B, but the incorporation of visual systems is expected to provide sufficient information (Fuentes-Pacheco et al., 2012). Furthermore, human actor handling generally makes a system quite error prone. Provided the data collection is based on sensor fusion and the algorithms used to stitch the output of several sources together are sufficiently robust, the error range arising from human actor handling can be minimized (Girard et al., 2011).

- The geometric context of an indoor environment is defined by the shape of a room, which can be generalized to the shape of its floor. How representation of a floor and its surrounding walls could be extracted from a polygonal mesh is discussed in this research.

- The mesh created is assumed to represent a (set of) room(s) on a single level. The influence of height differences on the quality of the scan as made with the available hard- and software is discussed in Appendix B.

- The floor library is assumed to be stored as `ifcSlab` standard case, which consists of a polyline describing the footprint of the floor, extrusion depth and direction, coordinates defining placement inside the BIM model, a unique id, and relevant semantic information (§ 4.3). A similar assumption is made about the storage of the walls extracted from the reference model.

- Matching the geometric position of an actor to the geometry of the model, allows for the actor to retrieve the correct semantics, as they are attached to the correct geometry in the model. Therefore, positioning an actor into the model is performed by comparing the geometry the actor captures of his environment to a library of match candidates. The generation of a representative library containing valid geometries is out of scope.

- The match itself is based on spectral graph theory. A comparison of graph matching techniques or the optimization of the process are left out of scope. Some elements which may influence the performance are mentioned.

The overall process that results in a mesh as result of performing VI-SLAM is defined by the fields described in the THEORETICAL FRAMEWORK (FIGURE 1.7). Appurtenant developments are out of scope of this research, though highly influential to the quality of the output. A more pertinent question is why a polygonal mesh as its output is a useful representation of indoor space, and what that means to the conceptual understanding of indoor localization. From this definition stems the approach to match permanent characteristics of indoor space. The creation of the mesh is handled at a data level, accompanied by the translation of what such a match would mean in the description of the indoor environment.

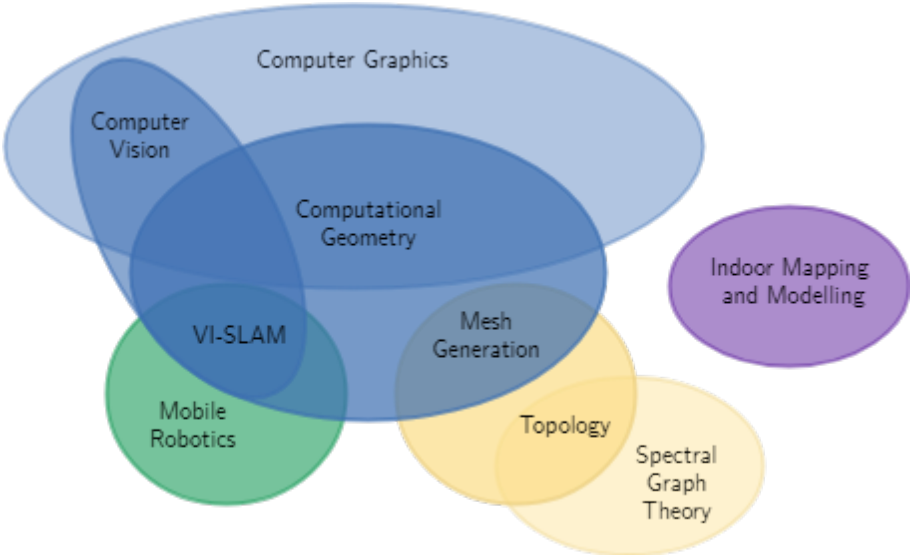


FIGURE 1.7: Euler diagram of fields that apply to the theoretical framework of this research.

1.6 Research Relevance

The main problem to be solved is to design an indoor localization process that can be made available on mobile devices. This is done by placing captured data inside a reference model as to position the capturing device. The information encapsulated into the reference model can then provide for location information, by retrieving the correct semantics. By displaying the information retrieved from the reference model, the method can help an actor understand their surroundings. Then a navigation path can be derived from the reference model and the heading can be determined using the Inertial Measurement Unit (IMU) of the mobile device. The SLAM techniques used to built a model for localization can be used to update said model in order to trace a reliable trajectory.

In designing this process, the core problem to solve is how to connect an actors current position with the appertaining location in the reference model. For solving this problem, a 3D representation of the indoor environment containing a set of positions relative to each other is available, as well as a 3D building model containing building elements, connectedness of elements and semantic information for each element. The desired output is the actors current position relative to building model, so that the relation to building elements is known and relevant semantic information can be retrieved. A solution is devised by aligning position model to semantic model by comparing the floor of the former to the set of floors embedded in the latter. As such, indoor localization is interpreted as a data matching problem. In other words, both input sources are analysed and adapted to a data format in which all information relevant for making a correct match is available. A choice is made to solve this problem from a graph point-of-view, as this allows for efficient and intuitive computation. Furthermore, a plethora of purely geometric approaches is already available. In stead of adding to possible geometric solutions, this research aims at exploring whether a different type of approach would be possible for retrieving pass-

able results. Looking at the indoor localization problem from a non-geometric standpoint may just result in insights necessary to further improve methods existing thus far.

The research entails the measurement and representation of indoor space as a structure embedding geometry, topology and semantics. The representation is assembled from point clouds and trajectory, together building a polygonal mesh. Correct interpretation of this geometric data results in a floor as the representation of indoor space, which can be compared to a reference set. The registration of the floor into the connected reference set builds a context, resulting in geometric information describing the indoor environment as a place with meaning.

This process entails the representation of 3D objects in indoor space, as well as their assembly. This information is then interpreted and reshaped , for efficient use in planning activities. The design of the full process adds to research & development in the field of indoor localisation, as it is based on a new combination of techniques. All of this according to what the field of Geomatics contains according to General Assembly of the International Federation of Surveyors (FIG) (Lemmens, 2011, p13). As for the MSc Geomatics course, data capture using Sensing Technologies is involved, as well as storage and analysis according to Geographic Information System (GIS) and 3D Modelling theory. Furthermore, the larger process is designed according to Location Awareness theory. Lastly, the execution is done using Python Programming.

Zlatanova et al. (2013) created an overview of problems to be solved in the field of IMM (FIGURE 1.8). All mentioned subjects from the **Acquisition and Sensors** category are of strong influence on the quality and interpretation of the data, though out of scope. The method designed aims at **Discovering the context of space** in a way that is useful for all research fields involved (see FIGURE 1.7). This ties in to the challenge Zlatanova et al. (2013) present as "to discover the context of environments automatically so that the devices used for acquisition and modelling become universal." Furthermore, the matching of a mesh model with a BIM overlaps with **Integration with GIS/BIM**. As concluded in this research, the differences in interpretation of what a space is strongly affect the possibilities of integration.

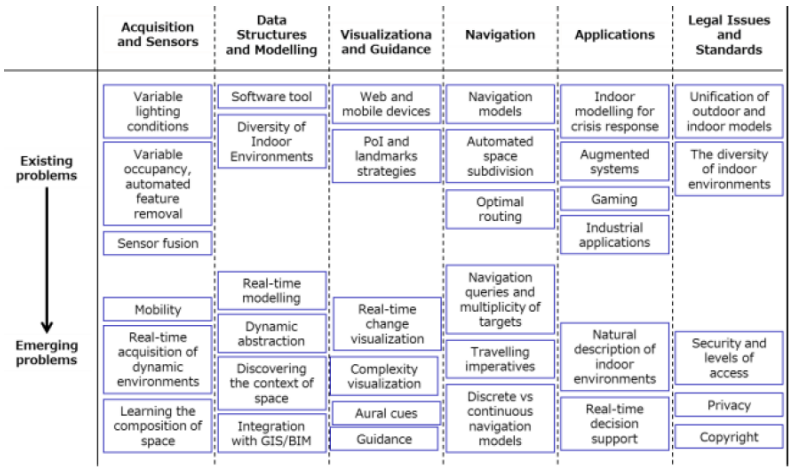


FIGURE 1.8: Overview of existing problems in IMM (Zlatanova et al., 2013).

To extract a graph from a model created on-the-fly and to connect it with a reference model containing semantic information allows for a plethora of possible applications. Some examples:

- Performing indoor localization.
- Performing indoor navigation based on the extracted graph.
- Enable accurate and specific LBS.
- Enrich Ambient Assistent Living (AAL) systems with location information.
- Enable medical applications such as patient and equipment localization.
- Enable up-to-date location information for rescue services in case of an emergency.
- Enrich BIM with furniture.
- Monitor changes in buildings compared to the designed state or the updated stated stored in BIM.
- Update existing BIM to an as-built state.
- Enable further data collection for construction and facility management according to the precise location within the building (quantitative).
- Enable further information collection for construction and facility management according to the precise location within the building (qualitative).
- Provide a more robust skeleton to built AR applications on, e.g. games, interactive experiences, retail augmentation, navigation.

1.7 Research Design

As defined in the PROBLEM DEFINITION, the possibilities of performing indoor localization in real-time on a mobile device are explored in both a theoretical and practical sense (FIGURE 1.9). The theoretical research is aimed at defining what characterizes indoor space, and results in a TERMINOLOGY definition of all related concepts, from an interdisciplinary viewpoint. If the defining characteristics of the context of indoor space are known in terms of concept and data types, data structures to support indoor localization can be identified. The THEORETICAL FRAMEWORK outlines concepts defining the full process, from which theoretical and practical output can be derived. The analysis of such a framework can lead to new insights in development, especially when a culmination of adjacent though often separately active research fields is required. Often, answers to problems in one field could be found in another, if only a mutual understanding of the problem definition can be reached. Contributions to be made to the discussion of theoretical concepts and thus the further development of a scientific field can be found in theory-oriented research (Verschuren & Doorewaard, 2010). Such a qualitative design is best served in an iterative manner, where concepts are examined in case studies. The

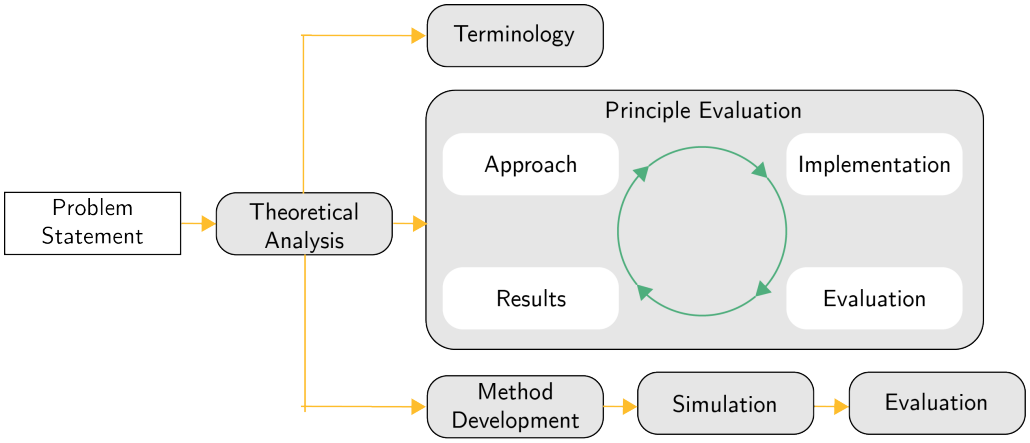


FIGURE 1.9: Research Design for exploring the possibilities of performing real-time indoor localization on a mobile device.

principles underlying this process are evaluated theoretically as well as practically. The core of the research is then aimed at the development of a METHODOLOGY for enabling indoor localization by providing data alignment. IMPLEMENTATION & RESULTS of the method simulation and principle evaluation are described and results are evaluated. Finally, CONCLUSIONS can be drawn and recommendation for further development of the full system are made.

2 | Terminology

This chapter introduces the main concepts concerning the presented research, and makes discernments in the meaning of terms in different fields. On the one hand, a single concept may be represented by diverging terms across disciplines, and on the other hand similar terminology may represent completely different concepts. Mathematics lays a common ground, a solid basis upon which many other theories can be built. As this field contains the most universal language as well, its terminology is taken as reference.

2.1 | Space

Intuitively, we may perceive space as what cannot be seen or touched, as what is left when every other instance around is defined. The term space has become an abstraction of everything outside of the boundaries of everything else. Ekholm and Fridqvist (2000) state that the general interpretation of a space is "an empty volume, enclosed in some respect - materially or experientially". Merriam-Webster (2019) defines space as "a limited extent in one, two or three dimensions", which can also be referred to as a **volume**.

In mathematics and philosophy, space has become what defines the boundaries of everything embedded within. A definition more fitting to this description would be "the unlimited expanse in which everything is located" (Wolfram | Alpha, 2019b) or "a boundless three-dimensional extent in which objects and events occur and have relative position and direction" (Merriam-Webster, 2019). A **space** represents a structure, often a set with relational properties, to which all its members adhere, or "a set of mathematical entities with a set of axioms of geometric character" (Merriam-Webster, 2019).

The characteristics of a type of space provide the context in which objects can be examined and represented. A type of space may inherit all the characteristics of a parent space, thus adding further restrictions to embedded elements. Thus, a space often represents a grid in which a world can be represented. Each volumetric partition of said grid not occupied by any element can intuitively be perceived as **free space**.

The space that completely envelops mathematical objects is called **ambient space** (Wolfram | Alpha, 2019a). This distinction is made since objects can also be isolated from ambient space, so that its intrinsic characteristics can be examined. The type of ambient space determines the shape and representation of an object.

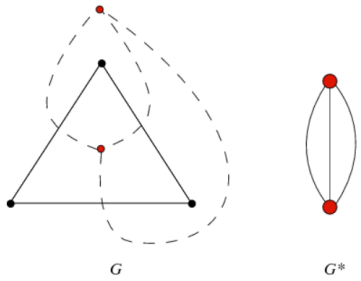


FIGURE 2.1: Graph $\mathcal{G}$ and its dual $\mathcal{G}^*$ (Weisstein, 2019a).

2.1.1 | Topological Space

Topology describes the relationships between (parts of) objects, that do not change under continuous deformation (Huisman & de By, 2009). As these relations are intrinsic, and thus invariant of ambient space, topology studies the construction of an object, rather than its shape. It can be used to "abstract the inherent connectivity of objects while ignoring their detailed form" (Weisstein, 2019c). A **Topological Space** is devoid of geometric structure, so that the relations and characteristics of objects can be examined qualitatively. The topology T on set X then defines topological space (X, T). Mapping an object into an ambient space may change its geometrical properties, but its topological properties remain.

The topology on set X, which contains a collection of open subsets $\{t\}$ where $t \in T$ as well as empty subsets ϕ, is defined such that the intersection as well as the union of any number of subsets is still open and part of T (Cromley, 1989; Edelsbrunner & Harer, 2010; Weisstein, 2019c). In a more geometric definition, at least one neighbourhood $\mathcal{N}$ can be defined for each point $x \in X$ (Worboys & Duckham, 2004). The intersection of any two neighbourhoods of x is in itself another neighbourhood of x.

Network Space

The topology of a set can be simplified to a graph structure $\mathcal{G}$, embedded in a **Network Space** that defines the topological connectedness between its members using an unordered set of node pairs. A **Graph** $\mathcal{G} = (\mathcal{N}, \mathcal{L})$ is a set of nodes pairwise connected by links which abstractly represent the connectivity between elements (Cromley, 1989; Nourian, 2018; Worboys & Duckham, 2004). The type of data represented by a graph is ambiguous and it does not need an ambient embedding. The dual $\mathcal{G}^*$ of a graph can be constructed by translating k-dimensional features to $n-k$-dimensional features in $\mathbb{R}^n$ (Nourian, 2016; Weisstein, 2019a). For example, a 2D face in $\mathbb{R}^2$ is mapped to its dual 0D node, and connected to another node only of the concurring faces are adjacent (FIGURE 2.1).

The dimensionality of a geometric graph structure is also called the topological dimension, and this combinatorial topology can be expressed by the Euler characteristic χ (EQUATION 2.1), using the number of faces $\mathcal{F}$, edges $\mathcal{E}$ and vertices $\mathcal{V}$ (Edelsbrunner & Harer, 2010). If two objects generalize to the same value of χ, while taking into account

the number of boundaries and holes i.e. geni, they are **homeomorphic**, meaning they would generalize to the same exact shape under topological transformation (Nourian, 2018).

$$\chi = \mathcal{V} - \mathcal{E} + \mathcal{F} \tag{2.1}$$

Manifold

A **Manifold** can be seen as a topological space, which locally resembles Euclidean space (Edelsbrunner & Harer, 2010; Rowland, 2019). Each point m of an n-dimensional manifold M has an open neighbourhood $\mathcal{N}$ that is homeomorphic to an open neighbourhood in $\mathbb{E}^n$. Such a subset U of M can be charted to $\mathbb{E}^n$ using a reversible coordinate transformation function φ, and a collection of charts can fully represent a manifold. The chart is then defined by ordered pair (U, φ). Lower dimension orientable manifolds can be fully represented in $\mathbb{R}^n$ space through a finite triangulation, and may be imposed with a metric for quantitative analysis.

2.1.2 | Metric Space

Metric Space is "a set with a global distance function, that for every two of the set's points gives the distance between them as non-negative real number" (Weisstein, 2019b), or the structure in which distances between all members can be defined (Worboys & Duckham, 2004). A metric, as ordered pair (S, d) can be imposed on a set S of arbitrary type, by defining how a shortest distance d between two members can be calculated. A structure is a complete metric when this shortest distance between two points is defined by a straight line, i.e. $d(\mathbf{s}, \mathbf{t}) = |\mathbf{t} - \mathbf{s}|, \mathbf{s}, \mathbf{t} \in \mathbb{R}^n$. Metric space has a natural topology based on the notion of proximity, as for each point a set of nearest neighbours can be found based on distance, and such a neighbourhood forms an open set (Munch, 2017; Worboys & Duckham, 2004).

Euclidean Space

The complete metric set of which all members are directly related in terms of distance and angle, is called the **Euclidean Space** $\mathbb{E}^n$ (Worboys & Duckham, 2004). It contains all n-tuples of real numbers (Stover & Weisstein, 2019) and provides for an intuitive abstraction of physical space without a specific origin. It defines the structure of the real vector space $\mathbb{R}^n$ of the same dimension[1].

When modelling 3D Euclidean space by real coordinates using $f : \mathbb{E}^3 \rightarrow \mathbb{R}^3$, each point is defined an ordered set of three coordinates $\mathbf{p} = (X, Y, Z)^T$, related to an arbitrarily defined origin (Huisman & de By, 2009). Each point can be related to each other point directly, using angle and distance.

[1] https://math.stackexchange.com/questions/831048/difference-between-euclidean-space-and-vector-space

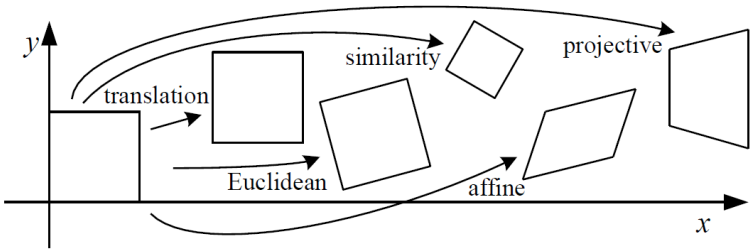


FIGURE 2.2: Hierarchy of coordinate transformations in $\mathbb{R}^2$ and $\mathbb{R}^3$ according to how many characteristics are preserved (Szeliski, 2010).

2.1.3 | Coordinate Mapping

Members of a space can be indexed using coordinates, by imposing a **coordinate frame** upon the structure. Each object can then be defined by coordinates, of which the ordering depends on the characteristics of the specific space. Each spatial property can then be accessed by sets of real numbers (Huisman & de By, 2009; Worboys & Duckham, 2004).

Coordinate systems can be used to map members of one type of space to another, using coordinate transformations (FIGURE 2.2). The type of transformation necessary fully depends on the characteristics of the objects to be mapped, and the type of relation that needs to be preserved (Marel, 2016; Szeliski, 2010; Worboys & Duckham, 2004). For each transformation, a function f can be built that maps set S to image T, so that each member of the original set relates to exactly one member of the created set, as $f : S \rightarrow T$.

Real Coordinate Space

Real Coordinate Space $\mathbb{R}^n$ then is a set of vectors, which represent all quantities that can be placed along a line. The coordinate of this value is denoted as a single real number in 1D-space, $\mathbb{R}^1$. By expanding this continuous 1D field to a second dimension, the coordinates of each value in this 2D space $\mathbb{R}^2$ consist of an ordered pair of real numbers, and so on. A real coordinate space is per definition metric, as each coordinate value measures the distance between a point and the systems' main axes.

Cartesian System

The position of an object in Euclidean space can be represented in a **Cartesian Coordinate System**, which represents a regular rectangular grid to map the earth (Barile, 2019; Huisman & de By, 2009). Though in essence its origin is arbitrary, it is most commonly chosen at the centre of the earth (geocentric), at a point on the surface of the earth (topocentric), or at the heart of a sensor (Marel, 2016).

Each position in a Cartesian system can be represented by the Cartesian product, i.e. a 3D position is described by an ordered set of three real numbers $\mathbb{R}^3 = \mathbb{R} \times \mathbb{R} \times \mathbb{R}$. Such a system is uniquely defined by a a rotation matrix $\mathbf{R}(\Omega_X, \Omega_Y, \Omega_Z)$ and a translation vector $\mathbf{t} = (t_X, t_Y, t_Z)^T$.

2.1.4 | Indoor Space

In order to design a method to teach a mobile device to understand the space around it, a clear definition of that space is necessary, in terms of concepts and data types. A plethora of definitions is available, however rarely useful in multidisciplinary work.

Often, a space is defined in terms of either *place*, *boundaries* or *function*. For example, Huisman and de By (2009) describe a geographic space as a *place* that has a position on the face of the earth. In the OGC CityGML standard, indoor space is inside one or multiple buildings, *bounded* by architectural components (Lee et al., 2016). The IFC standard (ISO 16739) mentions enclosure of an area, as well as the fact that a specific *function* can be executed in a space. This space is bounded either by a functional transition, or a geometrical border.

However, mathematically speaking, a space is a structure that defines the characteristics and representation of everything inside that space (§ 2.1). Thus a space is not determined by how it is bounded, but by how it defines what is inside. A single type of space representation does not exclude the possibility of mapping an object into another. The physical world may be represented in several dimensions of ambient space, as to capture its complexity.

Thus, Ekholm and Fridqvist (2000) set out to form a definition of space as a property applicable to Architecture, Engineering & Construction (AEC). They acknowledge that the concept of space has received the dual interpretation described before, leading to discrepancies in designing automated processes encompassing space. They limit their conclusion to a construction entity space, as "an aspectual unit based on a spatial view on the construction entity". Zlatanova et al. (2014) further mention that interpretations of indoor spaces become far more complex, and define indoor spaces as artificial constructs, designed and developed for human activities.

Forming a definition encompassing the conceptual and mathematical complexity of indoor space would lead to it defining every instance of influence to its meaning, embedding everything inside. The restriction *indoor* does define a boundary, not to the space, but to the elements of the physical world which may be embedded into indoor space.

A full representation of this indoor space would allow an actor to execute activities as positioning, localization and navigation (§ 2.2.2), which would require geometric, topological and semantic information (Isikdag et al., 2013). The complex indoor space can thus be modelled threefold (FIGURE 2.3): the geometric *place* can be modelled in Euclidean space (§ 2.1.2), with positions $(X, Y, Z)^T \in \mathbb{R}^3$. The connectivity and *boundaries* can be modelled in topological space (§ 2.1.1) as a primal 2-manifold surface $\mathcal{S}$ and its dual graph $\mathcal{G}^*$. The *function* of each element can be modelled in semantic space, as a set of characteristics embedded as attributes of each element. As such, a space can be occupied by elements, compliant to the rules of the structure. The inherent organization has a great influence on how a representation of indoor space can be built, and on how it relates to the physical world. Understanding these implications by mathematically defining the type of models at hand, can aid in understanding how encountered problems can be solved.

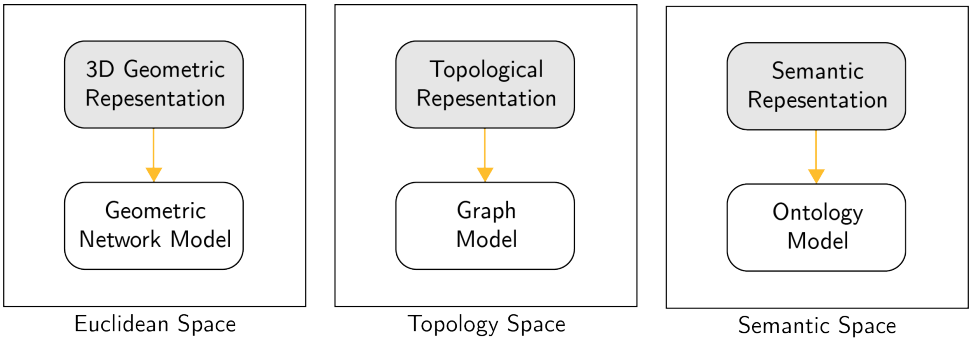


FIGURE 2.3: Interrelated models required for indoor positioning, localization and navigation. Above the primal representation required in its subsequent space, below each dual model. Based on Isikdag et al. (2013).

2.2 | Mapping Indoor Space

A **context** is the structure that describes how data is collected, stored, interpreted and used (Worboys & Duckham, 2004). Collection results in **data**, as measured numerical values that represent an object of study, or different entities of interest (Munch, 2017; Worboys & Duckham, 2004). Embedding elements of said data into a context results in a translation to **information** (Worboys & Duckham, 2004). A starting point is often a **viewshed**, as a map of all points visible from a location (Worboys & Duckham, 2004).

2.2.1 | Mapping the Physical Environment

A physical **environment** can be simplified into a **model**, in which each **object** embedded into the environment is represented by data points. A geometric representation of these objects results in a **map** of the environment.

The formal representation of an object as dictated by its ambient space is found in **geometry** (Worboys & Duckham, 2004). Thus an object is placed in a **structure**, which arranges elements and their relationships (Huisman & de By, 2009; Marel, 2016). The type of structure objects are placed in depends on the embedding type of space, and determines nomenclature of elements (TABLE 2.1). Physical objects are mostly placed into Euclidean space, and can be simplified to piecewise linear geometrical objects. Abstract generalisation of said objects can be made using differential geometry, which can be transformed into a topological structure. Algebraic topology decomposes the local object structure, which can be further abstracted using Graph Theory. As such, an indoor environment can be modelled directly using geometry and topology.

Elements of a structure can be indexed, by imposing a coordinate system upon it (Worboys & Duckham, 2004). The physical environment is most often represented in Euclidean space, upon which a geometric **system** can be imposed, which continually defines the distance between the full extent of an element and the system origin (Botsch et al., 2010; Huisman & de By, 2009; Marel, 2016). Similarly, **Geodesic** or **Spherical** coordinates are used to map spherical space, as the angle of rotation of an element with respect to the system origin (Huisman & de By, 2009; Marel, 2016). A **coordinate transformation**

can be executed in order to chart elements of spherical space to Euclidean space, and vice versa. Similarly, a 2-manifold surface can be **charted** to parametric space, by transforming the surface coordinates (Botsch et al., 2010). The mathematical similarity is seen in the assumption that a 2-manifold can be represented in $\mathbb{R}$, as can a spherical object, while locally resembling 2D Euclidean space. Thus, the representation of a physical environment can be placed into a physical context, as well as simplified in order represent information of interest only.

The interpretation of geometric objects is executed by attaching **semantics**, or to study its meaning Merriam-Webster (2019). The definition of a meaning diverges strongly per field, according to the boundary conditions for object definition. In IMM, **semantics** is meaning given to objects, as a clear definition of building elements as well as their properties and functional states, and as usage of spaces (Isikdag et al., 2013).

2.2.2 | Positioning, Localization & Navigation

If data consists of numerical values and context translates it to information, such a transcendence can be made in indoor applications specifically as well. Pure **positioning** entails finding a point or area occupied by a physical object or person (Groves, 2013; Mautz, 2012). This term is most used as to find a position $\mathbf{p} = (X, Y, Z)^T \in \mathbb{R}^3$, relative to the systems' origin. Contrasting to the quantitative position, a location entails finding a qualitative description of the occupied area (Groves, 2013). Providing semantic context for a position would then allow an actor to find a location with a specific meaning, or to perform **localization**. It requires topological correctness of sensors used to determine the underlying position, and precise accuracy of is generally of less importance (Mautz, 2012). Based on either position or location, an actor can execute **navigation**, as determining heading and velocity of the current trajectory, and being guided along an optimal path to the destination (Groves, 2013; Mautz, 2012).

N-D	Euclidean Geometry	Piecewise Linear Geometry	Differential Geometry	Algebraic Topology	Graph Theory
0D	Point	Point	Point	Vertex	Node
1D	Line	Line-Segment	Curve	Edge	Link
2D	Plane	Polygon	Surface	Face	Cycle
3D	Hyper-Plane	Polyhedron	Solid	Body	Clique

Table 2.1: Terminology for object members in several geometrical and topological fields, according to the dimensionality of the represented object. Based on Nourian (2018).

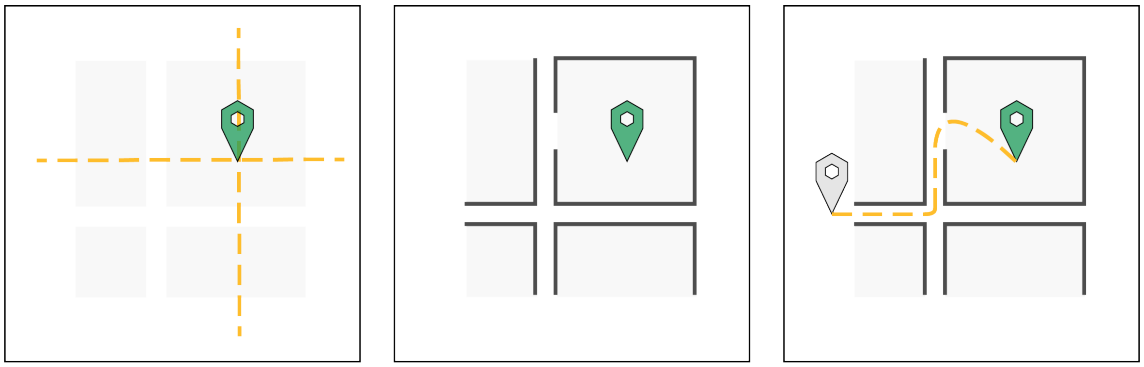


FIGURE 2.4: Differences between positioning, localization and navigation for indoor situations.

3 | Theoretical Framework

The following chapter describes the theoretical framework underlying real-time indoor localization on a mobile device. IMM is introduced as a multidisciplinary field (§ 3.1), which aims at aligning common practices from adjacent fields in order to solve problems in the complex indoor environment. As such, extensive research into indoor positioning methods (§ 3.1.2) has led to a solution from mobile robotics and computational geometry, SLAM (§ 3.2). Similarly, using computer vision, this practice has expanded to performing VI-SLAM as a method to perform the process as autonomously as possible (§ 3.3). As the process requires the storage of both position of elements and nearness, it is common to save its geometry as well as its topology, e.g. in a polygonal mesh (§ 3.4). The topological structure can be analysed using spectral graph theory (§ 3.5). The relations between the mentioned fields are visualized in FIGURE 1.7.

3.1 | Indoor Mapping and Modelling (IMM)

Indoor Mapping and Modelling (IMM), in itself a multidisciplinary field, has benefited from technological advancements in many disciplines, e.g. sensor development, computer vision, robot navigation and 3D visualization (Zlatanova et al., 2013). While primarily developed as providing tools for professionals, mainly in the AEC sector, a growing commercial interest in location based information has instigated an expansion upon the field, as to create IMM models, such that they can become widely attainable and available (Worboys, 2011). IMM requirements are now widened to incorporate facility management applications as well, though most requirements and uncertainties applying to existing buildings have not been considered yet (Volk et al., 2014). However, the increasing use of applications requiring indoor models in new buildings instigates a call for applicability in existing buildings as well. The adoption of these requirements in IMM for existing buildings implies a large set of use cases.

A solid manner of representing the indoor environment is found in BIM, as a means of representing geometry, topology and semantics, along with the possibility to connect additional required information to relevant elements (Hichri, Stefani, Luca, & Veron, 2013; Isikdag et al., 2013; Zlatanova et al., 2013). The incorporation of these interrelated information layers results in a rich model fit for many indoor applications (FIGURE 2.3). The primal layers should be directly stored and allow for retrieval and supply of information as well as visualization. The dual layers may be extracted for efficient computation and provide information on geometrical, topological and semantic relations.

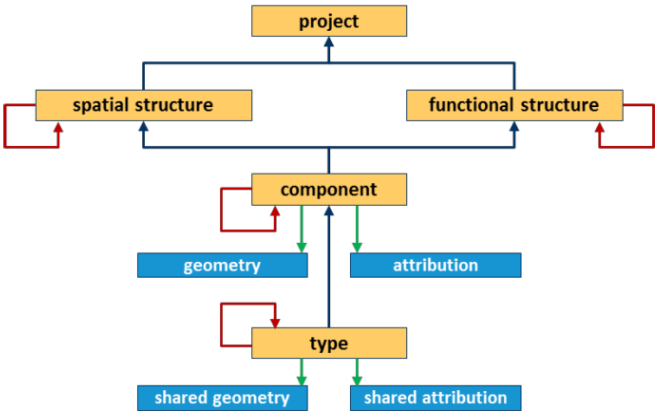


FIGURE 3.1: General breakdown structure of an IFC model (Borrmann et al., 2017).

3.1.1 | Building Information Model (BIM)

BIM was created to represent both form and function of a building, stored as geometry and semantics, to provide for an information resource in all processes describing the life cycle of a building (BuildingSMART, 2016). It is built upon the open standard IFC (ISO 16739), which allows for encoding and transporting necessary data and information.

The Industry Foundation Classes (IFC) was instigated as an initiative to facilitate interoperability in the building sector, as to speak the same language industry wide. The standard allowed the AEC sector to digitally define building and infrastructure elements as rich objects, accompanied by semantics and further informational attributes. The integration of many standards and definitions has lead to a plethora of incorporated attributes, and the possibility to simultaneously store an object under different types of definitions. In an effort to prevent incomplete models, set of elements was defined as integral.

The overall structure of the data model is divided into spatial and functional definitions (FIGURE 3.1), where each component is represented by its geometry and semantics, and the type class of objects it belongs to. A type of spatial structure relevant to IMM is the class `ifcBuilding`, a required instance of the `ifcProject`. Central elements are then required to be stored as `ifcBuildingElements`, which contains a set of central elements to bound indoor space (§ C.1). The empty volumes in between may be defined as `ifcSpace`, representing a function imposed within a single or multiple rooms.

As such, the definition of geometric representation of building elements attributed with semantics builds a rich 3D model for IMM. Furthermore, a certain degree of closeness of elements may be extracted from the natural topology of the 3D Euclidean space the geometry is mapped in (§ 2.1.2). For many objects the placement is defined as relative to a related object, and some attributes are available for direct storage of topological information as well, as part of the class `ifcRelationship`.

Most available BIM models represent buildings in an as-planned state, as the model was generated as part of the design process (Ptrucean et al., 2015; Volk et al., 2014). However, a growing interest in as-built models requires extensive research into efficient

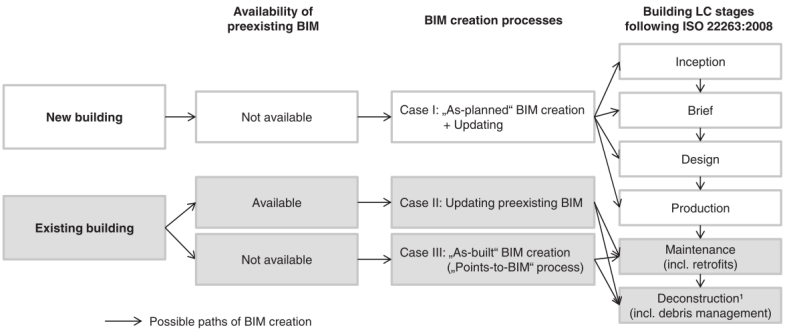


FIGURE 3.2: Classification of BIM creation processes (Volk et al., 2014).

and precise modelling of existing buildings, directly requiring further development of IMM methods. Much research is taken into more and more automated scan-to-BIM processes, e.g. (Armeni et al., 2016; Ochmann, Vock, Wessel, & Klein, 2016; Thomson & Boehm, 2015; Zeibak-Shini et al., 2016). However, little research is taken into updating existing models, though integral to the use of the model throughout the building lifecycle (FIGURE 3.2).

As part of the envisioned technical applications for BIM, the localization of building elements remains a problem to be solved (Costin, Pradhananga, & Teizer, 2012; Volk et al., 2014; Zhou, Ding, Wang, Truijens, & Luo, 2015). Current research leans toward the adoption of pure positioning methods like RFID, and manually finding relative coordinates within the model.

3.1.2 | Indoor Positioning Methods

Having outdoor positioning working smoothly, indoor positioning started as an extension thereupon, using similar systems (Yassin et al., 2016). The Global Navigation Satellite System (GNSS) radio waves used in outdoor positioning cannot penetrate walls, and thus are rendered useless in indoor environments. The principle of Radio Frequency Identification (RFID) positioning should be adoptable, using radio waves from terrestrial access networks, e.g.cell towers. Though the principal has proven to work for close-range positioning, accuracy remains low and significant errors still arise (Mautz, 2012; Yassin et al., 2016). The great length of radio waves makes it difficult to achieve accuracy on an indoor level. In an attempt to resolve these issues, systems combining different radio signal positioning methods have been designed. However, the need for accurate time synchronization in these systems leads to costly constructions, and are therefore not fit for consumer use.

Indoor positioning methods have thus been expanded beyond using radio waves, towards other types of sensor networks, e.g. Global System for Mobile Communications (GSM), Universal Mobile Telecommunication System (UMTS), or wireless communication using Long Term Evolution (LTE) on a long range, and Bluetooth or Wireless Local Area Network (WLAN) on a short range. Though a much higher accuracy can be reached using these networks, they cannot always be available everywhere. Further-

more, the specific installation of an indoor positioning network can again be costly, and therefore not viable as a broadly implemented system. In order to circumvent additional investments, these systems could be based on existing sensor networks. However, the accuracy of systems designed for maximum coverage in stead of indoor positioning is not sufficient either, especially for floor identification (Bot, Braaksma, Braggaar, Ligtvoet, & Staats, 2016; Mautz, 2012).

As opposed to relying on contingent systems, a device can utilize its built-in Micro Electro Mechanical Systems (MEMS) to perform relative positioning autonomously (Lemmens, 2013; Yassin et al., 2016). Such inertial systems have proven to be capable of returning accurate results. However, in consumer applications low-cost MEMS are used, again deteriorating the reliability of the output. Though many noise reducing algorithms are available, the regained accuracy is not deemed sufficient for indoor positioning.

Promising solutions are found in the hybrid fusion of different sensors, e.g. IMU and cameras (Mautz, 2012). Especially on (sub)room level, required accuracy for indoor applications can be reached by utilization of Red-Green-Blue-Depth (RGB-D) imagery .

Theoretically, the hybridization of autonomous and contingent systems would result in a most optimal solution, though Mautz (2012) state that reliance on context information from a 3D building model would mean no network is necessary to reach the same level of accuracy and coverage.

3.2 | Simultaneous Localization and Mapping (SLAM)

In the field of mobile robotics, SLAM is executed in order to perform navigation by positioning a robot in real-time, while building a context around it. Simultaneous Localization and Mapping (SLAM) encompasses the realisation that localization and mapping operations are best executed simultaneously, as both enhance the quality of the other (Fuentes-Pacheco et al., 2012). It must be noted that the definition of localization in mobile robotics is devoid of context, thus entails a positioning process (§ 2.2.2) and will further be referenced as such.

The SLAM process requires the robot to track its position $\mathbf{x}_k$ relative to former positions, the heading $\mathbf{v}_k$ chosen from each position, and the relative position $\mathbf{m}_i$ of each landmark in sight (Bailey & Durrant-Whyte, 2006). The position and heading at a certain moment in time comprise the pose $(\mathbf{x}_k, \mathbf{v}_k)$ of the robot. The set of these vectors form a geometric map and trajectory in $\mathbb{E}^3$ space. Each collected **landmark** is an edge or corner point easily detectable by a camera from every angle, as it strongly contrasts with its environment (Fuentes-Pacheco et al., 2012). Indoor, door and window frames as well as floor edges generally fit this purpose.

3.2.1 | Robot Navigation

Robot navigation is comprised of different categories (FIGURE 3.3), all of which contribute to an accurate and usable representation of the real world (Stachniss, 2009). *Mapping* is the combination of gathered data describing the surroundings, integrated into a useful representation for real-time positioning and navigation. *Localization* is then pose estimation, relative to a map or data model. Using only the map collected in real-time, this pro-

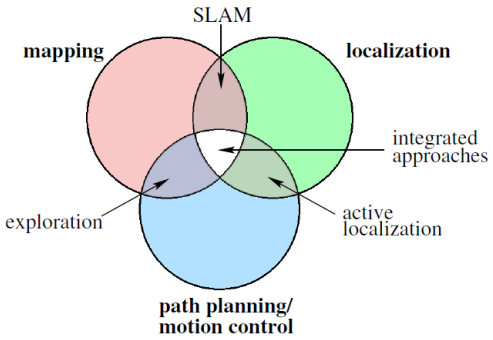


FIGURE 3.3: Concepts of robot navigation (Stachniss, 2009).

cess entails positioning (§ 2.2.2). *Path planning* entails the determination of free space and the navigation through it. Different combinations of these tasks have been researched, in order to solve the robot navigation problem. The culmination of all three is called Simultaneous Planning, Localization and Mapping (SPLAM). At the end of this process, a robot is expected to have built a full map of the environment, as well as have determined its pose inside it and being capable of navigating through it.

3.2.2 | Computer Vision

The science of computer vision entails attempts to create systems as capable of imaging and interpreting an environment as human vision (Szeliski, 2010). However, the computer is generally left with insufficient contextual information for recovering the unknowns in environment interpretation. What is to be interpreted is usually so complex, researchers are still aiming to reach the depth of knowledge necessary to understand each aspect of our every day environment (Zlatanova et al., 2013).

Computer vision has proven to successfully recreate geometric information, enriched with attributes that can be gathered using light (Szeliski, 2010). Generally, Red-Green-Blue Imaging (RGB) cameras are used to capture e.g. colour and illumination, but the additional use of Infra Red (IR) in RGB-D cameras has increased the range of collectible information. However, perception of what is captured remains a seemingly impossible task. Promise may lie in a combined system, where more complex steps in information interpretation are left to humans, after the computer has analysed all it can, using the limited amount of contextual information it can gather.

3.2.3 | Visual Inertial Simultaneous Localization and Mapping (VI-SLAM)

Much research has gone into the improvement of SLAM systems, by using different types of data sources in order to obtain more accurate results (Fuentes-Pacheco et al., 2012). Over time, a tendency has arisen towards vision based systems, as these are capable of obtaining the widest range of information using a single type of sensor. Cameras can perform object detection and recognition, as well as obtain information about range

and the appearance of the environment. Furthermore, they are attainable, affordable, lightweight, portable, low on power consumption and intuitively appealing (Riisgaard & Blas, 2004).

In order to improve visual SLAM systems, the output can be enhanced using Visual-Inertial Odometry (VIO). This visual-inertial approach was created to increase accuracy and robustness of the system. Further enrichment to VI-SLAM can be provided by stereo configurations, which provide easy and accurate calculation possibilities for the 3D positions of detected landmarks, through triangulation.

A core step in VI-SLAM is the detection of keyframes, as a captured image or video frame containing a sufficient amount of landmarks and different enough from the former (Fuentes-Pacheco et al., 2012). They are used to efficiently estimate the pose of the sensor and to reduce information redundancy.

Still, visual systems are fairly error prone. Fuentes-Pacheco et al. (2012) have listed known issues in visual systems, as much of the research takes many of these issues for granted due to the heavy weight of the benefits. A first setback is the limited range of visual systems, rendering VI-SLAM for external and large scale environments too great a challenge in many cases. As visual systems depend on feature detection, environments with too many or too little salient features become difficult to handle. Visual repetition in an environment causes mismatches, resulting in sudden displacement in the map, or erroneous corrections to it. A dynamic environment causes the VI-SLAM algorithm to register too little known features to perform accurately, which leads to unpredictable errors. Purely from a sensor point of view, erratic movements and occlusion of the camera lead to errors as well. The movement of the camera is expected to be smooth and consistent, which can be controlled when the camera is carried by a robot. To some extent, the issues arising from human sensor handling can be solved using keyframes and actor training.

As concluded in § 3.1.2, providing a hybrid and integrated solution incorporating the input from several sensors would increase the performance of a positioning system (Groves, 2013). From a taxonomy of positioning methods perspective (FIGURE 3.4), VI-SLAM does incorporate as much information that can be drawn from environmental features as possible.

3.2.4 | Related Work

Some interesting recent developments in the field of SLAM have been published by Dos Santos et al. (2016); Gálvez-López, Salas, Tardós, and Montiel (2016); Lopez-Antequera, Petkov, and Gonzalez-Jimenez (2016); Moteki, Yamaguchi, Karasudani, and Yoshitake (2016); Sattler, Weyand, Leibe, and Kobbelt (2012).

3.3 | Performing VI-SLAM on a mobile device

As explained in § 3.2, the process for performing indoor positioning with VI-SLAM entails utilizing a devices' visual system (§ 3.3.1) and IMU (§ 3.3.2) in order to simultaneously perform mapping (§ 3.3.3) and positioning (§ 3.3.4) tasks. The following explores how such a process would be executed, to determine how its functionality can be examined.

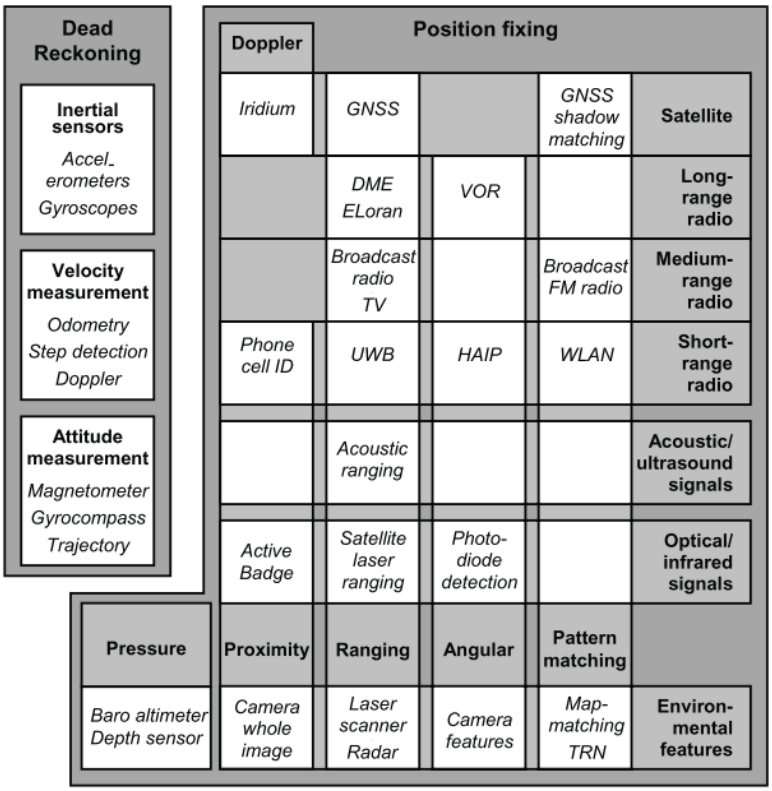


FIGURE 3.4: Taxonomy of positioning principles (Groves, 2013).

3.3.1 | RGB-D Cameras

In the capture of close range environments, passive stereo vision has been accumulated into best practices for a long time, despite the inability of these systems to capture difficult scenes, containing few features or reflective surfaces. Not considering lighting conditions, cameras are capable of gathering sufficient information to geometrically model the environment (Szeliski, 2010). Other well known problems may result in measurement errors, e.g. sudden movement of the sensor. Errors caused by actors could be limited, by applying direct training, through remarks that the actor should scan sharp edges from different angles and preferably with a rotated camera also, as to increase the accuracy of landmark detection (Zafari et al., 2017).

Konolige (2010) has proven that the combination of structured light patterns and stereo vision is capable of handling challenging scenes. The same problem has also been addressed using time-of-flight systems by Henry, Krainin, Herbst, Ren, and Fox (2010a). The culmination of different environment capturing approaches can be provided in RGB-D cameras, which are often turned into active sensors by the incorporation of an IR emitter. The images captured by these cameras store depth values per pixel, in addition to the colour. The depth then defines the distance between the image plane and the object a pixel represents. From these 2.5D images, a frame-by-frame point cloud is created

FIGURE 3.5: A frame by frame point cloud created using a VI-SLAM process.

(FIGURE 3.5), which is stitched together by the SLAM algorithm used.

Another drawback to vision-based systems is the limited Field of View (FOV) cameras generally provide. This problem might be solved by the integration of a fisheye camera, in order to capture a broader partition of the scene, also providing for stereo vision. However, image distortion is much stronger in this type of camera, requiring solid and active camera calibration (Remondino & El-Hakim, 2006). Access to the camera's intrinsic parameters allows for a useful evaluation of suitability for the performance of VI-SLAM. Calculations to relate the captured image to the real world can be made using EQUATION 3.1, according to the intrinsics **K** in FIGURE 3.6. It further includes focal lengths (f_x, f_y), preferably in pixels, and image skewness s.

$$\mathbf{K} = \begin{bmatrix} f_x & s & x_0 \\ 0 & f_y & y_0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.1}$$

Most accurate results will stem from applying sensor fusion (Fuentes-Pacheco et al., 2012; Mautz, 2012; Rebolj, Pucko, Babic, Bizjak, & Mongus, 2017; Yassin et al., 2016; Zafari et al., 2017), where the interpolated depth value then represents the most accurate depth for a single pixel. However, the integration of all of these high quality sensors may result in a hefty and costly solution for consumer grade devices. Concessions could be made by including sensors of less quality, or by utilizing e.g. either stereo vision or depth sensing. The growing attention for visual systems in home entertainment and gaming has pushed the development of these devices towards high quality gear for consumer grade prices (Huang et al., 2017).

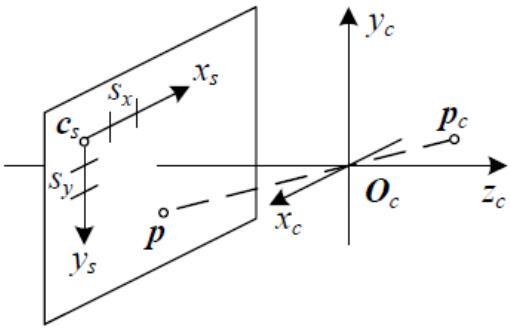


FIGURE 3.6: Intrinsic camera parameters, with principal point or optical centre (x_0, y_0) as the translation from the actual image plane origin, $\mathbf{c}_s$. The projection centre or nodal point $\mathbf{O}_c = (X_C, Y_C, Z_C)$ marks the camera coordinate frame origin, accompanied by the camera constant C as distance between the principal point and the projection centre, and (s_x, s_y) are pixel spacings (Szeliski, 2010).

3.3.2 | Trajectory Tracking

Endres et al. (2012) state that in order to smoothly execute SLAM using RGB-D, the camera trajectory should be estimated concurrently to the creation of a 3D model. The estimated pose graph then increases the quality of the model created, an implementation that works well on devices not including an integrated IMU. The estimation of a trajectory is part of the SLAM process according to Bailey and Durrant-Whyte (2006), as the current pose and heading are constantly stored (§ 3.2). Thus, the trajectory and the model of the environment can be updated an corrected simultaneously, as a way to improve the accuracy of the process.

The keyframes captured by a visual system can be used as an input to computing a trajectory, using the cameras extrinsic parameter matrix $[\mathbf{R}|\mathbf{t}]$ (FIGURE 3.7). These parameters lie at the basis of the coordinate transformation applied to relate the image to real world coordinates (FIGURE 2.2), and can be used to perform position calculations based on methods in photogrammetry (Lemmens, 2011), using EQUATION 3.2.

$$[\mathbf{R}|\mathbf{t}] = \left[\begin{array}{cccc} R_{00} & R_{10} & R_{20} & t_X \\ R_{01} & R_{11} & R_{21} & t_Y \\ R_{02} & R_{02} & R_{22} & t_Z \end{array}\right] \tag{3.2}$$

A more reliable trajectory is composed of tracking of positions over time by all incorporated sensors (Mautz, 2012). Then, gross errors, e.g. sudden movement, can be accounted for. By comparing both trajectories, or at least the points a trajectory contains, drift correction can be applied. On the small scale it corrects misalignment between concurrent point cloud frames (FIGURE 3.5), stitching them together as precisely as possible. On the larger scale, drift in the trajectory itself can be corrected when a sufficient amount of landmarks is recognized that should be placed at different position than currently apparent from the trajectory (FIGURE 3.8).

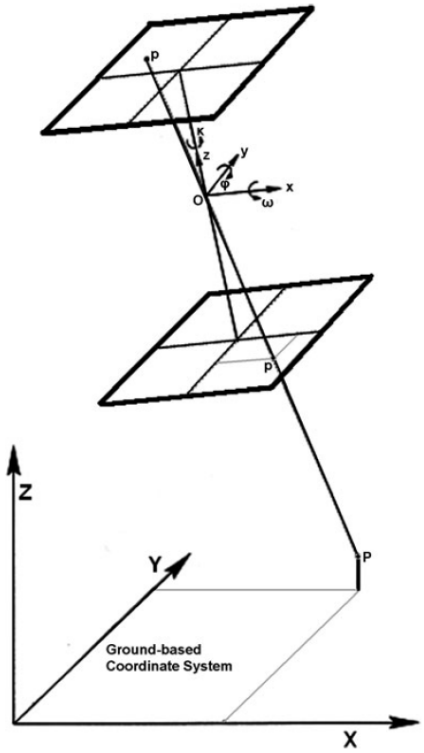


FIGURE 3.7: Extrinsic camera perimeters, with real-world coordinates of object $\mathbf{P} = (X, Y, Z)$, real-world coordinates of the camera projection centre $\mathbf{O} = (X_w, Y_w, Z_w)$ and Euler rotations around the projection centre $(\omega, \varphi, \kappa)$ (Lemmens, 2011). The Euclidean difference between $\mathbf{P}$ and $\mathbf{O}$ is transformation vector $\mathbf{t} = (t_X, t_Y, t_Z)^T$ and the rotations can be generalized to rotation matrix $\mathbf{R}$.

3.3.3 | Mapping

As explained in § 3.2, the SLAM process positions an actor while simultaneously building a map around it. Bailey and Durrant-Whyte (2006) present this map as containing time ordered sets of vehicle positions $\mathbf{x}_i$, control points $\mathbf{u}_i$, environment landmarks $\mathbf{m}_n$ and landmark observations $\mathbf{z}_i$. Many different algorithms have been developed in order to build a reliable map and trajectory from these observation sets. In the field of mobile robotics, such a purely geometric map may provide sufficient information. However, for a coherent representation of indoor space, more information is needed (§ 2.1.4).

So far in 3D IMM, the topological information is extracted from 2D floor plans and assumes that rooms are empty. However, a 2D plan layering approach is very prone to errors (Zlatanova et al., 2014). Many plans are not up to date, or represent the designed structure rather than the current state. An effort could be made to update floor plans to an as-built state, though this effort would not solve the problems at hand. A more viable solution would be to fully create 3D models of the indoor environment. This is generally done by capturing a point cloud of the environment, from which its geometry can be extracted. The model can then be used to extract topological information from.

Another method would be to build a topological structure simultaneously to the geo-

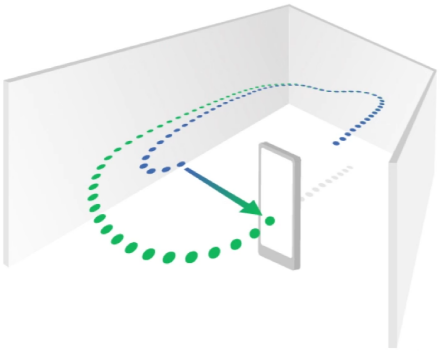


FIGURE 3.8: Large scale drift correction applied to improve the quality of the output model. Here, the stored trajectory is shown (blue), which is corrected to the actual trajectory (green) after recognizing a landmark placed earlier in the loop (Google, 2017b)

.

metrical one. This would require an integrated process, continuously updating both the geometry and the topology. The algorithms performing a SLAM process could be utilized to build this topological structure alongside the geometry, therefore enriching the created map to an information model. Such a model could be stored as a polygonal mesh (§ 3.4), and be created and updated on-the-fly (FIGURE 3.9).

The captured data is prepared for mesh storage, by applying constant transformations. Successive point cloud frames are stitched together, as images (Szeliski, 2010). First, the correspondence between each frame is sought, as a matching of features apparent in both keyframes compared. Second, the transformation parameters describing positional differences are determined (§ 2.1.3), as the difference in extrinsic parameters from both frames. Third, transformations are applied in order to align the frames. Most often this is done using bundle adjustment, as the extrinsic parameters for a large collection of datasets is adjusted simultaneously. Fourth, the aligned frames are composited. The transformation generalizes to $f : \mathbf{p} \rightarrow \mathbf{p}'$, where in the least the intrinsics $\mathbf{K}$ and extrinsics $[\mathbf{R}|\mathbf{t}]$ are applied.

In essence, the rotation is described as a set of Euler rotations. These axis-angle rotations are sensitive to the order of execution, and thus unstable (Szeliski, 2010). A general axis-angle description may be much more useful, as EQUATION 3.3, using rotation θ around axis $\mathbf{e}$ and identity matrix $\mathbf{I}$. However, the outcome may be ambiguous. A most stable rotation representation would be a quaternion $\mathbf{q} = (x, y, z, w)$, describing rotations on the unit sphere $||q|| = 1$, where each member of the quaternion describes an axis relation (FIGURE 3.10).

$$\mathbf{R}_n = \mathbf{I} + sin\theta \times \mathbf{n} + (1 - cos\theta) \times \mathbf{n}^2 \tag{3.3}$$

3.3.4 | Positioning

While executing the SLAM process, a current position is constantly updated according to the built map, continuously updating the trajectory. If executed autonomously the

FIGURE 3.9: Polygonal mesh of the scene in FIGURE 3.5, created on-the-fly.

process functions as dead reckoning (FIGURE 3.11), where further movement from the trajectories origin results in ever increasing errors (Groves, 2013; Mautz, 2012). In order to correct this, a place recognition method should be implemented, in order to correct the current position, and thus the trajectory accordingly.

In SLAM, large scale loop closure is performed in order to account for drift in position tracking (FIGURE 3.8), where correspondence is sought between image (keyframes) and map (landmarks) (Williams et al., 2009). The environment is scanned, its features are compared to the data already stored, and if a match reliable enough occurs the position of the sensor is compared to the positions of the matched features. If a drift above a certain threshold is found, it can be corrected for in the algorithm. Thus a place, as a collection of objects in an environment, can be recognized by computing correspondence in storage of said objects.

3.4 | Triangular Meshes

A triangular mesh is a discrete model of topological space, containing a data structure to store both the geometrical and the topological component of a surface.

3.4.1 | Surface Representation

In geometry processing a surface most often represents an orientable continuous 2-manifold (§ 2.1.1) describing the boundary of a 3D solid (Botsch et al., 2010). Boundary δS of solid S then defines the difference between *inside* and *outside*. For digital representation and processing purposes, the 2-manifold is mapped from topological space (X, T) into $\mathbb{R}^3$. As the mapping operation is executed as a coordinate transformation, the resulting represen-

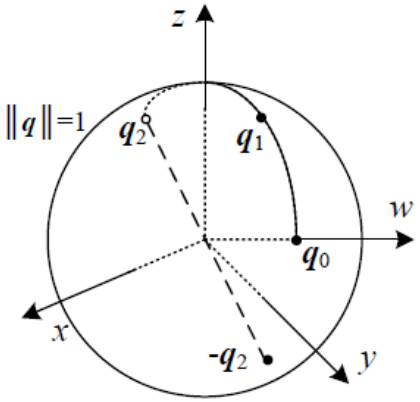


FIGURE 3.10: Unit sphere $||q|| = 1 \in \mathbb{R}^4$, which describes $\mathbb{R}^3$ rotations, using parameters $\mathbf{q} = (q_x, q_y, q_z, q_w)$ (Szeliski, 2010).

tation can only be an approximation of the actual surface. Depending on the application, surfaces can be approximated in an *implicit* or a *parametric* manner.

Implicitly, such a surface $\mathcal{S}$ is defined by the zero-set of function F, so that every 3D point $\mathbf{x}$ on the surface becomes part of the boundary manifold embedding a solid shape (EQUATION 3.4). However, the surface may be disconnected. Implicit representation can be handled piecewise by dividing the space encapsulating the surface, most commonly as tetrahedrals or hexahedrals (i.e. voxels).

$$F : \mathbb{R}^3 \to \mathbb{R} \qquad \mathcal{S} = \{\mathbf{x} \in \mathbb{R}^3 \mid F(\mathbf{x}) = 0\} \tag{3.4}$$

The *parametric* charts of a 2-manifold can be defined by a vector-valued function $\mathbf{f}$, and is composed of ideally connected 2D differential forms Ω (EQUATION 3.5). In essence, Ω symbolizes the local resemblance of $\mathbb{E}^2$ in the 2-manifold, but is often generalized to larger portions for data size reduction. The shape can be processed efficiently by handling members of Ω separately, which are commonly defined as triangles or quadrangles.

$$\mathbf{f} : \Omega \to \mathcal{S} \qquad \Omega \subset \mathbb{R}^2, \quad \mathcal{S} = \mathbf{f}(\Omega) \subset \mathbb{R}^3 \tag{3.5}$$

3.4.2 | Mesh Representation

Botsch et al. (2010) define a polygonal mesh as a piecewise linear surface representation, which consists of a *topological* and a *geometric* component. In its essence it is a parametric surface representation. A 2-manifold surface as topological structure can properly be embedded into $\mathbb{E}^3$ if it is orientable, thus representing the boundary of one or more volumes (Edelsbrunner & Harer, 2010). For a mesh to represent a 2-manifold surface, each edge should be incident to only one or two faces, and the faces incident to a vertex should form a closed or open fan (Nourian, 2018).

The *topological* component consists of vertices $\mathcal{V}$, faces $\mathcal{F}$ and sometimes edges $\mathcal{E}$, which together define the structure of mesh $\mathcal{M}$ as a simplical complex. Each embedded

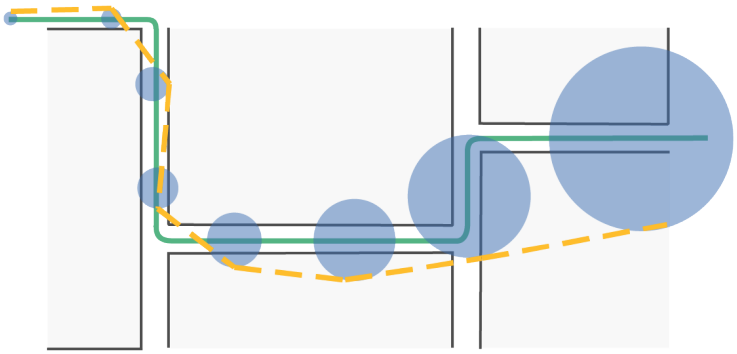


FIGURE 3.11: Error accumulation in autonomous real-time positioning process. Green represents the real trajectory taken through a hallway. The accumulated error over time is denoted by the blue circles, generalizing to a possible trajectory in yellow.

simplex Δ^2 can be defined by three vertices, three edges and a single face. Each edge is an element of two combined ordered vertices, and each face an element of three combined ordered vertices. The topological component of $\mathcal{M}$ is denoted in EQUATION 3.6.

$$\begin{aligned} \mathcal{V} &= \{v_1, ..., v_{\mathcal{V}}\} \\ \mathcal{E} &= \{e_1, ..., e_{\mathcal{E}}\}, \quad e_i \in \mathcal{V} \times \mathcal{V} \\ \mathcal{F} &= \{f_1, ..., f_{\mathcal{F}}\}, \quad f_i \in \mathcal{V} \times \mathcal{V} \times \mathcal{V} \end{aligned} \tag{3.6}$$

The topological structure of a mesh can be mapped to $\mathbb{R}^3$ by assigning coordinate values to each vertex, as in EQUATION 3.7. Now, each vertex $v_i \in \mathcal{V}$ is associated with a position $\mathbf{p}_i \in \mathbb{R}^3$ by mapping all vertices $\mathcal{V}$ to points P, as the *geometrical* part of the mesh

$$P = \{\mathbf{p}_1, ..., \mathbf{p}_i\}, \quad \mathbf{p}_i := \mathbf{p}(v_i) = \begin{pmatrix} X(v_i) \\ Y(v_i) \\ Z(v_i) \end{pmatrix} \in \mathbb{R}^3 \tag{3.7}$$

3.4.3 | Mesh Processing

The generation and processing of surfaces as triangular meshes follows a well established pipeline (FIGURE 3.12), constructed to generate the most optimal visualization possible (Botsch et al., 2010). Here, input data is used to generate a mesh, in which errors are detectable. The type of input data dictates some typical artefacts which require repair, and for each type topological errors and inconsistencies should be removed, in order to generate a proper 2-manifold surface. Surface quality is improved by these repairing operations, as smoothing and fairing operations would. Analysis of the quality in this stage predicts necessity and application of such operations, which handle noise as well. Then,

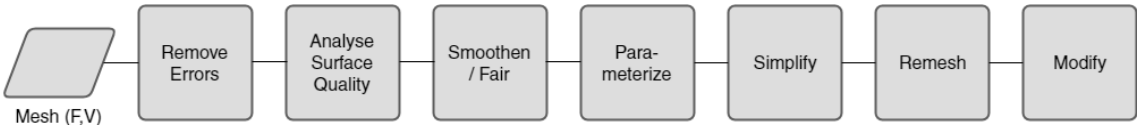


FIGURE 3.12: Flowchart of geometry processing pipeline, based on Botsch et al. (2010).

a parametrization can be created on the mesh, as mapping the manifold into a $\mathbb{R}^3$ coordinate system and charting the full extent into $\mathbb{E}^2$. Its output may serve many functions, i.e. appearance-preserving simplification. To simplify a mesh is to reduce its complexity without diminishing the quality of the output. As such, computational complexity of further operations is reduced. Then, the quality of non-topological properties is improved by remeshing algorithms, after which it can be freely modified according to application specific requirements.

Many of these operations rely on finding the rate of change within the mesh itself, as to divide a mesh where adjoining triangles behave very differently, and smoothen it where adjoining triangles are defined in a strongly similar matter (Botsch et al., 2010). In other words, the intrinsic tendency of partitions of the surface to change direction dictates the interpretation of (the quality of) the surface. The places in the mesh to clean, smoothen, cut, simplify or remesh are found using the Laplacian operator Δ. It is defined as the divergence of the gradient of a surface, i.e. $\Delta = \nabla^2 = \nabla \cdot \nabla$.

3.5 Spectral Graph Theory

Spectral graph theory is based on the notion that the structural properties of graphs can be intuitively examined using the eigenvectors and eigenvalues of matrices based on such a graph Chung (1996); Luxburg (2007); Nourian, Rezvani, Sariyildiz, and van der Hoeven (2016); Spielman (2007). The combinatorial information which can be extracted gives a direct analogy between manifold geometry and graphs (Chung, 1996). It provides efficient and relatively simple solutions to complex clustering and analysis problems, often faster than their direct counterparts.

Given an unweighted graph $\mathcal{G}(N, \mathcal{L})$ with nodes N and links $\mathcal{L}$, adjacency information can be written in matrix form $\mathbf{A}_{i,j}$ as EQUATION 3.8. The degree $\mathbf{D}_{i,j}$ of such a graph is the diagonal matrix representing the number of adjacent nodes d_i for each node n_i.

$$\mathbf{A}_{i,j} = \begin{cases} 1 & if \quad (i,j) \in \mathcal{L} \\ 0 & otherwise \end{cases} \tag{3.8}$$

Even more graph properties can be extracted from the Laplacian $\mathbf{L}_{i,j}$, which is found by $\mathbf{L}_{i,j} = \mathbf{D}_{i,j} - \mathbf{A}_{i,j}$ and defined as EQUATION 3.9. This Laplacian operator $\mathbf{L}$ behaves as the Laplacian Δ would on a mesh surface (§ 3.4.3). Its spectrum of eigenvalues $0 = \lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_n$ allow for a simplification as $\mathbf{L}\mathbf{x} = \lambda\mathbf{x}$, corresponding with eigenvectors $\mathbf{x}_1, \ldots, \mathbf{x}_n$. The eigenvector of a system is the vector that remains unchanged under linear transformation, only scaled by its corresponding eigenvalue. In other words, multiplying matrix $\mathbf{L}$ by eigenvector $\mathbf{x}$ yields the same result as scaling the eigenvector by its eigenvalue λ. The smallest eigenvalue defined as $\lambda_1 = 0$ applies no linear transformation and

thus corresponds to an eigenvector $\mathbf{x}_1 = \mathbf{1}$ of all ones. The number of eigenvalues which return $\lambda_i = 0$ furthermore corresponds with the number of connected components in the graph.

$$\mathbf{L}_{i,j} = \begin{cases} d_i & if \quad i = j \\ -1 & if \quad (i,j) \in \mathcal{L} \\ 0 & otherwise \end{cases} \tag{3.9}$$

3.5.1 | Spectral Graph Embedding

A spectral embedding of a graph can be given by assigning eigenvectors as coordinate scalers to node i, e.g. placing it at point $(\mathbf{x}_2(i), \mathbf{x}_3(i))$. The eigenvectors are used to draw an almost always planar embedding, uniquely representing the underlying structure of the graph (Luxburg, 2007; Nourian, 2016; Spielman, 2007). The exact result however can be influenced by the choice of matrix on which the eigenvectors are based.

Spectral graph embedding is realized by fixing some nodes to the corners of a convex polygon, which requires the rest to be placed at the average position of its neighbours. The choice for nodes to fix is rather random, and a different choice would only result in a rotated version of the same embedding, as the scaling applied by the eigenvector is invariant to rotation in itself (Spielman, 2007). As such, the spectral embedding of similarly shaped graphs will return similarly, though perhaps rotated. The larger the eigenvectors used for this embedding, the further the nodes will appear from their neighbours.

3.5.2 | Spectral Clustering

The second smallest eigenvalue, λ_2, provides for a particularly interesting interpretation of a graph. The graph is only connected if $\lambda_2 \geq 0$, and the further from zero, the stronger the graph is connected. Its corresponding eigenvector $\mathbf{x}_2$, the Fiedler vector, gives information about connectivity (Chung, 1996; Luxburg, 2007; Spielman, 2007). The sign of its values may be interpreted as a division of the data into parts according to its sparsest cut, or the cut for which the least edges are broken. A cut between disjoint sets $A, B \subset \mathcal{N}$ where the similarity between a pair of nodes (i, j) is defined as s_{ij} can be stated as EQUATION 3.10. Alternatively, the weight w_{ij} of a link may be used. The quality of such a cut is defined by the number of edges to break, which optimizes to as few as possible. Such cuts may be utilized for clustering a graph.

$$cut(A, B) = \sum_{i \in A, j \in B} s_{ij} \tag{3.10}$$

A threshold t can always be found such that the set of edges in a cluster $S = \{i : \mathbf{x}_2 \leq t\}$ is most optimally connected. A common solution is to find the median cut as $n_{left} < 0 < n_{right}$, meaning to find the largest gap between values of $\mathbf{x}_2$ within the graph (Spielman, 2007). A clustering algorithm can be constructed which optimizes the cuts to be made, in order to divide a graph into clusters (Luxburg, 2007). The type of Laplacian used at the basis of cut definition determines the exact results of the clustering algorithm.

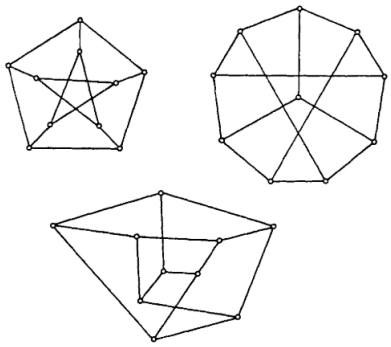


FIGURE 3.13: An example of three isomorphic graphs (Read & Corneil, 1977).

Alternatively, a spectral embedding of graph nodes may be used as input for a clustering algorithm. The spectral properties of the graph then determine the placement of each node as data point. Within this placement clusters can be determined, based on which the original dataset can be divided. The embedding chosen as well as the type of clustering algorithm applied then determine the results.

3.5.3 | Graph Isomorphism

Isomorphism of two objects is defined as "having the same form" or structural properties (Weisstein, n.d.-a). If graph $\mathcal{G}$ and graph $\mathcal{H}$ contain the same number of vertices connected in the same way, they are said to be isomorphic, i.e. $\mathcal{G} \cong \mathcal{H}$ (Weisstein, n.d.-b). $\mathcal{H}$ then forms a one-to-one mapping of $\mathcal{G}$, such that adjacency is preserved (Read & Corneil, 1977). Isomorphism is independent of embedding (FIGURE 3.13), thus purely representative of the underlying structure of compared graphs.

The complexity of solving graph isomorphism is theoretically unsolved, though many practical applications have been developed (Dawar & Khan, 2018). The theoretical analysis of Read and Corneil (1977) concludes that the type of graph to be compared highly determines what type of algorithm would be best suited to solve graph isomorphism.

3.5.4 | Graph Matching

Graph matching entails finding correspondence between two given graphs, where the underlying structures are at least highly similar (Conte, Foggia, Sansone, & Vento, 2004). Testing isomorphism is an example of finding an exact match. Using real data, exact matching is often too rigid, due to e.g. noise or processing steps. By relaxing some of the constrains to matching graphs, inexact matches can be retrieved.

Spectral graph matching then forms a basis to find structural correspondence between graphs (Conte et al., 2004; Spielman, 2007). As the spectral map of a graph is independent of vertex ordering, using eigenvectors limits the extent in variation. Basically, if the eigenvalues of $\mathcal{G}$ and $\mathcal{H}$ are different, the graphs are different. However, graph attributes cannot be incorporated in this method.

4 Methodology

Performing indoor localization on a mobile device is interpreted as a data matching problem, where a scan of an environment is compared to a reference model (§ 1.1). The scan is the result of a SLAM process (§ 3.2), and its execution results in a mesh created on-the-fly (§ 3.3). It is referenced to a BIM (§ 3.1.1), in order to find the current position relative to the contextual model. This relation determines the semantics describing the current environment, transcending a position into a location (FIGURE 1.4).

The input represents a real-time environment of the actor, and a reference library that contains information on the environment. To create a connection between a scan and reference model is to interpret the environment an actor is in. If the room or corridor is recognized, semantic information can be retrieved accordingly. As the shape of the floor and its surrounding walls determine the general shape of a room, the characteristics of these building elements should be recognizable. After extracting a proper room description from the scan, it is cross-referenced to a library of rooms contained within the building. A match returns a unique room id, which allows the user to retrieve the appertaining semantics. A general workflow can be found in FIGURE 4.1.

First, the input sources are analyzed in § 4.1. Then follows a projection of a 3D model to a 2D graph. The output of this projection should be similar enough in type, such that a match can be made. As both input models are based on different data types derived from different interpretations of indoor space, both require a different projection method. In order to form the mesh $\mathcal{M}(\mathcal{F},\mathcal{V})$ into graph $\mathcal{G}(\mathcal{N},\mathcal{L})$ a method is designed in § 4.2. For forming the solids $\mathcal{S}(c,d)$ extracted from the BIM into graph $\mathcal{H}(\mathcal{N},\mathcal{L})$, a slightly different method is constructed in § 4.3. Then, a method is presented to form the match between $\mathcal{G}$ and $\mathcal{H}$ in § 4.4.

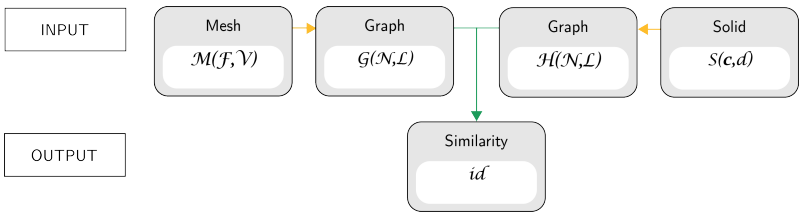


FIGURE 4.1: Methodological workflow for creating a match between a mesh created on-the-fly and a reference BIM library of solids. Both input types are projected into network space (yellow arrows) so they can be represented as graphs. These graphs are compared and matched based on similarity (green arrow).

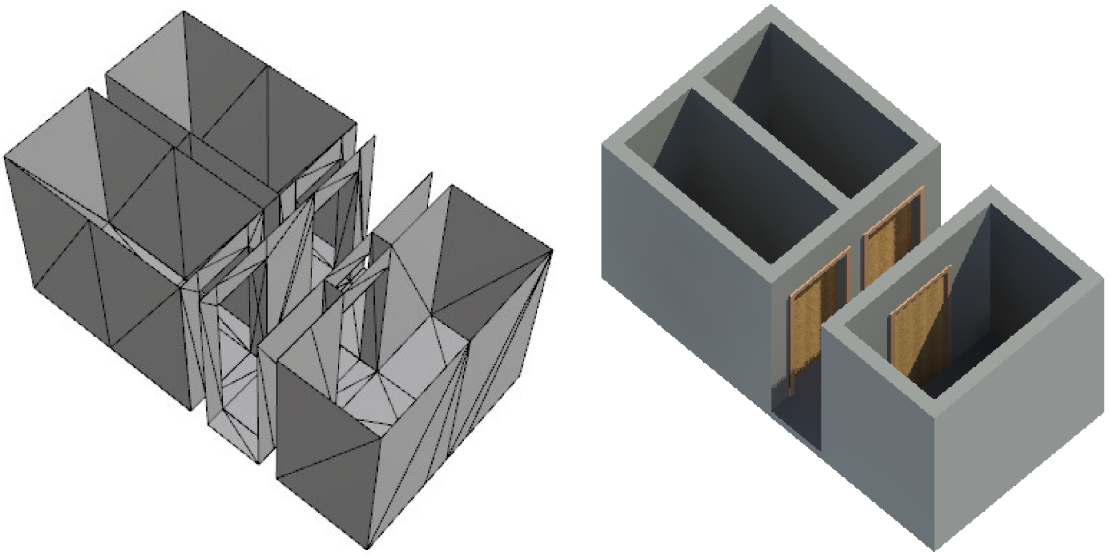


FIGURE 4.2: Test data used to visualize the methodology for matching a mesh (left) and a BIM (right) model representing the same indoor environment.

4.1 | Input Sources

The methodology for extracting a graph from both reference models and matching them is described in this chapter, and visualized using a test dataset (FIGURE 4.2). As the real data would, all data the method is tested on consists of a mesh and BIM model representing the same (set of) room(s) stored according to their respective data types. As FIGURE 4.1 implies, the mesh is stored using faces and vertices, and the BIM geometry as solids containing curves and depths.

4.1.1 | Mesh

The input mesh (§ 3.4) is built by executing a VI-SLAM process (§ 3.3) on a mobile device, and is stored in a .obj format. The storage is structured as an indexed face-set (TABLE 4.1). This means that all faces are stored according to their topological definition (EQUATION 3.6), and all vertices according to their geometrical components (EQUATION 3.7). Each vertex is accompanied by its vertex normal, and both are stored in counter-clockwise order per face, which makes the surface orientable. The normals are computed along with the creation of the mesh, in order to store face orientation based on visibility. The front faces thus represent the inside of a room, and back faces point outwards. Each vertex is further accompanied by texture information, based on which a coloured model can be displayed. A mesh model can be defined as a piecewise linear surface representation (§ 3.4.2), where each triangular face forms a small planar partition of the manifold surface it represents. This topological surface (§ 2.1.1) forms a draping of the actors' viewshed along a traversed trajectory. In other words, the mesh model stitches a blanket of everything the mobile device could see while scanning.

$\mathcal{V}$	
id	unique id
v	position [3]

N	
v_i	vertex id
n	normal [3]

$\mathcal{F}$	
id	unique id
v_i	vertex/normal id
v_j	vertex/normal id
v_k	vertex/normal id

TABLE 4.1: Data structure for an indexed face set. Each vertex $v_i \in \mathcal{V}$ is stored with a position $\mathbf{p}_i \in \mathbb{E}^3$, and accompanied by a vertex normal $\mathbf{n}_i$. Each face $f_i \in \mathcal{F}$ is defined by the id of its three adjacent vertices, in counter-clockwise order.

4.1.2 | BIM

The contextual model is stored as IFC BIM (§ 3.1.1), is assumed to be complete, and to represent an as-built state. As representative information model it contains geometry, topology and semantics (§ 3.1). The geometry is stored in the spatial structure of the model embedded in $\mathbb{R}^3$, which implies a natural topology. The functional structure allows for storage of topology and semantics inside attributes. The geometry type and specific storage differs per element. This architectural interpretation of a building is decomposed into building elements. In other words, all physical boundaries defining indoor space are explicitly defined, as well as additional elements that belong to the construction and installation of a building. The topological structure of a BIM is implied by the natural topology of the Euclidean space it is embedded in (§ 2.1.2). Furthermore, the main class `Relation` allows for the storage of several types of relational or topological properties of elements.

4.1.3 | Input source handling

Both input sources are handled separately in order to extract rooms from the model. The extraction methodology is based on the interpretation of an indoor environment each model type is based on. Room defining elements are extracted in order to build a dual graph of each model, consisting of connected clusters. Each cluster then represents a room, and each cluster connection a possibility to traverse from one room to another. Finally the characteristics of the graph are extracted on which a match can be based. Differences in interpretation of both models is summarized in TABLE 4.2.

The methodology for extracting a rich graph from both models as part of the main methodology referenced in FIGURE 4.1 is executed in three main steps (FIGURE 4.3). First, the graph itself is extracted as a full topological representation of the modelled indoor environment. Depending on the characteristics of the input source, either the graph is extracted then filtered and clustered based on room definition, or the rooms are extracted then translated to a topological graph. In both cases the resulting graph represents a network of rooms. Its characteristics can be analysed by computing the spectral map.

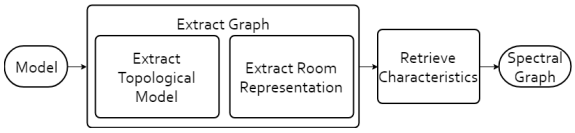


FIGURE 4.3: Methodology for extracting a rich graph from a reference model.

4.2 | Graph Extraction from Mesh

The approach to extracting initial graph $\mathcal{G}$ from the mesh model is based on the storage and mathematical definition of the model (§ 4.1.1). Then $\mathcal{G}$ is prepared for matching by simplifying the model based on a general interpretation of indoor space (§ 2.1.4) into $\mathcal{G}'$. The steps can be found in FIGURE 4.4.

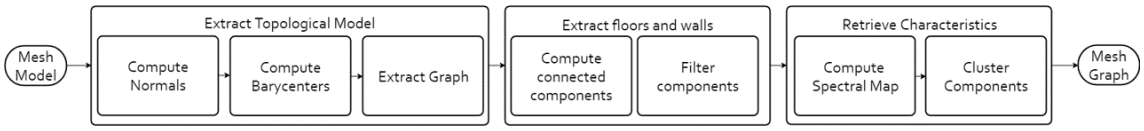


FIGURE 4.4: Method for extracting a graph from a mesh model.

4.2.1 | Compute Normals

In creation of the mesh, the direction of gravity as output of the IMU is taken into account. Though the point of origin remains at the pose of the device at the start of the scan, it is rotated so that the up-direction aligns with the corresponding axis. Thus, all normals directed upwards represent horizontal surfaces, a quality which can be utilized in order to extract the floor from the mesh. However, not every horizontal surface can be part of the floor. Assuming the actor was instructed to focus on scanning the extents of the floor, the height at which the largest amount of points are placed should be the height of the floor. As such, it can be separated from e.g. tables, cabinets and ceilings, although the

MESH	BIM
Viewshed of traversed trajectory	Collection of elements describing the construction of a full building
Topological structure embedded into metric space	Geometric structure embedding topological attribution
Geometry described by set of points	Geometry described by set of solids
Topology directly stored as indexed face set	Topology implied by embedding metric space, and stored as additional attributes
Semantics not included	Semantics stored as attributes
Captured on-the-fly	Built manually or semi-automatically

TABLE 4.2: Conceptual comparison of a mesh interpretation vs. a BIM interpretation of a model of indoor space.

latter are usually not well represented in scans taken on device with a limited FOV.

The normals of each face are computed and utilized for classification of connected components, according to similarity to neighbours $\mathcal{N}$ and a unit vector $\mathbf{k}$ directed upwards. The normal vector $\mathbf{n}$ is computed according to Newells method (EQUATION 4.1), as a robust basis for surface orientation. It separately computes the X, Y and Z components of the normal vector $\mathbf{n}$, traversing all vertexes of a face in order. Each face defining vertex v_i is decomposed into positions X_i, Y_i and Z_i. A cumulation of current vertex i and next vertex $i+1$ determines the output values.

$$\begin{aligned} n_x &= \sum_{i=0}^{n-1} (Y_i - Y_{i+1})(Z_i + Z_{i+1}) \\ n_y &= \sum_{i=0}^{n-1} (Z_i - Z_{i+1})(X_i + X_{i+1}) \\ n_z &= \sum_{i=0}^{n-1} (X_i - X_{i+1})(Y_i + Y_{i+1}) \end{aligned} \tag{4.1}$$

4.2.2 | Compute Barycenters

The barycenter of each face is computed mainly for visually representative mapping of the dual graph. It defines the placement of a dual node embedded in $\mathbb{E}^3$, as well as the anchor point for the face normal vector. Every point $\mathbf{p}$ on a surface can be barymetrically represented as in EQUATION 4.2 by parameters (α, β, γ) in relation to corner points $(\mathbf{a}, \mathbf{b}, \mathbf{c})$. The barycenter is found by mapping each parameter to $\frac{1}{3}$.

$$\mathbf{p} = \alpha\mathbf{a} + \beta\mathbf{b} + \gamma\mathbf{c} \tag{4.2}$$

4.2.3 | Extract Graph

As a mesh is a representation of topological space embedded into Euclidean space, both components can be extracted for analysis. As described above, the geometric component is utilized to determine surface orientation as a basis for segmentation and mapping. Such a model may then be clustered based on similarity and proximity of faces, forming implicit neighbourhoods $\mathcal{N}$. Alternatively, the topological component of a mesh can be utilized to form explicit neighbourhoods based on adjacency of faces. The dual graph of such a model (FIGURE 2.1) forms an efficient representation of the topological component, based on which structures and relations can be analysed.

A dual graph can be built according to a face to face adjacency list, which forms a barycentric subdivision of the surface. Each primal face is translated into a dual node, and each pair of adjacent faces determines a link between these nodes. In other words, two nodes are connected by a link if the corresponding faces are adjacent (Edelsbrunner & Harer, 2010; Worboys & Duckham, 2004).

Theoretically, two faces are accepted as adjacent, when they have two vertices in common. As the faces are expected to form a proper manifold surface, no faces would overlap

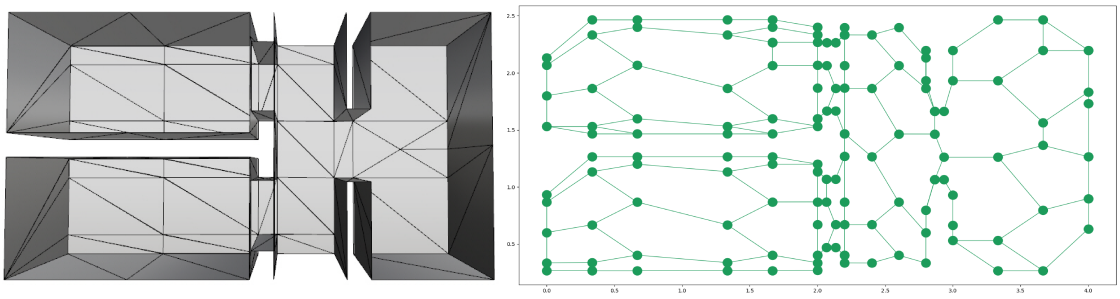


FIGURE 4.5: A mesh model and its dual graph $\mathcal{G}$.

or be disconnected. Thus face to face adjacency is determined by overlapping edges. A mapping of dual elements can be found in TABLE 4.3. This dual mapping results in a graph representation of the input mesh (FIGURE 4.5).

4.2.4 | Connected Components

The clustering of mesh elements is done by performing a search through the dual graph to find connected components, as the sets of subgraphs that are connected (Edelsbrunner & Harer, 2010). The process is performed by picking a semi-random seed and performing a depth-first search through its neighbours. Thus, the algorithm finds the connected subgraphs, by forming a full subgraph before finding a next one. A seed is selected if it has not been traversed yet, and adheres to a threshold for dot product similarity with the unit vector pointing upwards, $\hat{\mathbf{k}} = (0, 0, 1)^T$. Neighbours are selected according to adjacency stored in the graph structure and accepted if similar enough to the seed. The similarity s_{ij} is defined as the dot product of current seed i and current neighbour j, and tested to a threshold t_{sim}.

The connected components algorithm in ALGORITHM 1 returns all horizontally and vertically oriented clusters in the model, and attributes each link with the degree of similarity. The output is shown in FIGURE 4.6.

4.2.5 | Filter Components

The filtering of graph $\mathcal{G}$ is executed based on the output of the connected components algorithm. Faces which are oriented neither vertically nor horizontally are skipped by the algorithm. In the end, all components smaller than a threshold are filtered out. Fur-

PRIMAL MESH		DUAL GRAPH	
0D	Vertex	2D	(Cycle)
1D	(Edge)	1D	Link
2D	Face	0D	Node

TABLE 4.3: Mapping of a primal mesh to its dual graph. Each element not explicitly stored is written in brackets.

Algorithm 1 CONNECTED COMPONENTS

Input: Graph $\mathcal{G}$, Similarity threshold t_{sim}
Output: Set of graph links $\{l(i,j)\}$

function DFS($\mathcal{G}, t_{sim}$)
  $\{A\} \leftarrow$ nodes in $\mathcal{G}$
  while $\{A\}$ is not $\emptyset$ **do**
    if Current seed $\{S_c\}$ is $\emptyset$ **then**
      $\mathbf{n}_A \leftarrow$ normal of A_0
      check_up $\leftarrow$ unit vector $\hat{\mathbf{k}} \cdot \mathbf{n}_A$
      if check_up $> t_{sim}$ **then**
        $\{S_c\} \leftarrow A_0$
      end if
      $\{A\} \leftarrow \{A\} \setminus A_0$
    end if
    for i in $\{S_c\}$ **do**
      $\{\mathcal{N}_i\} \leftarrow$ Neighbours of i
      $\mathbf{n}_i \leftarrow$ normal of i
      for j in $\{\mathcal{N}_i\}$ **do**
        if j in $\{A\}$ **then**
          $\mathbf{n}_j \leftarrow$ normal of j
          check_up $\leftarrow$ unit vector $\hat{\mathbf{k}} \cdot \mathbf{n}_j$
          check_sim $\leftarrow \mathbf{n}_i \cdot \mathbf{n}_j$
          if check_up > t_sim **and** check_sim > t_{sim} **then**
            $\{A\} \leftarrow \{A\} \setminus A_j$
            $\{S_c\} \leftarrow \{S_c\} \cup A_j$
            $\{l\} \leftarrow \{l\} \cup (i,j)$
          end if
        else if j in $\{S_c\}$ **then**
          $\mathbf{n}_j \leftarrow$ normal of j
          check_up $\leftarrow$ unit vector $\hat{\mathbf{k}} \cdot \mathbf{n}_j$
          check_sim $\leftarrow \mathbf{n}_i \cdot \mathbf{n}_j$
          if check_up > t_{sim} **and** check_sim > t_{sim} **then**
            $\{l\} \leftarrow \{l\} \cup (i,j)$
          end if
        end if
      end for
      $\{S_c\} \leftarrow \{S_c\}\, i$
    end for
  end while
end function

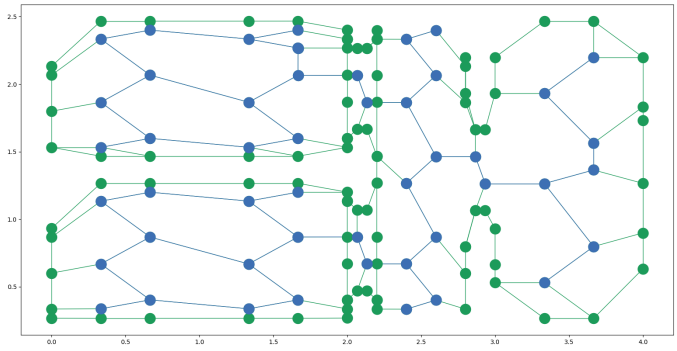


FIGURE 4.6: Graph $\mathcal{G}$ segmented into the largest horizontal component as floor (blue) and all large enough vertical components as walls (green).

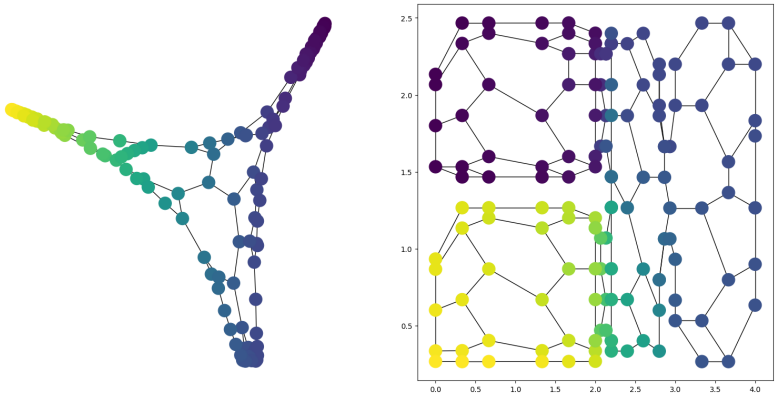


FIGURE 4.7: Visualization of spectral properties of graph $\mathcal{G}$. A spectral embedding (left) compared to positional embedding (right), both coloured by its Fiedler vector. Both visual representations prove an implicit occurrence of three or four clusters within the test data, of which the two rooms on the left are most strongly disconnected from the rest of the model.

thermore, a check is run in case of a horizontally oriented face. As only the largest one should represent a floor, this is the only one kept.

4.2.6 | Compute Spectral Map

The intrinsic properties of a fully connected graph can be extracted by computing its spectral properties (§ 3.5). A spectral embedding using its eigenvectors $(\mathbf{x}_2(i), \mathbf{x}_3(i))$ on node i gives an almost always planar and unique portrayal of the graph or the topological structure it represents (FIGURE 4.7). It symbolizes clusters within the graph and the degrees between these clusters, as well as the separate elements.

4.2.7 | Cluster Components

The Fiedler vector (§ 3.5.2) specifically implies the connectivity of the graph and can be used as a basis for clustering. The values it assigns to each node represent a division into

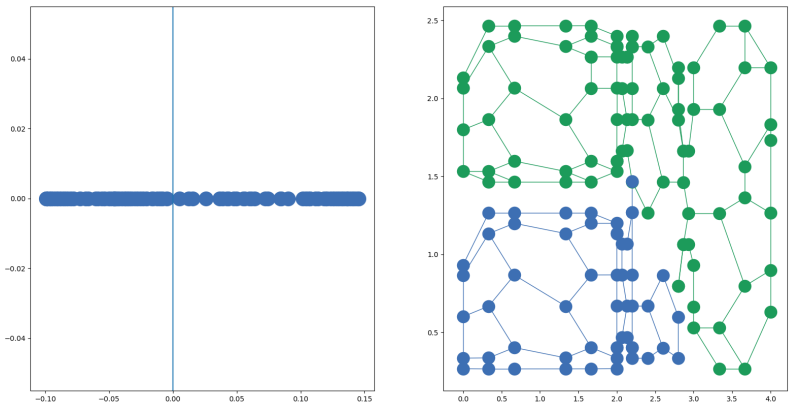


FIGURE 4.8: Sparsest cut in $\mathcal{G}$ based on the largest eigengap.

natural clusters within the graph, and the larger the gaps between these values are, the less connected the nodes are. As such, the Fiedler vector may be used to create cuts in a graph (FIGURE 4.8). A series of sparsest cuts may result in a set of graph clusters.

Alternatively to using Fiedler cuts to divide a graph, the spectral properties can be used as a basis for clustering. When mapping all nodes into a spectral embedding, the new coordinates imply clusters based not only on nearness of the faces all nodes represent, but also the connectivity between them. Applying a clustering algorithm on these data points returns implicit graph clusters. Here, two unsupervised clustering algorithms are applied, in order to compare results.

To separate the data into clusters of equal variance according to their mean value, a k-means algorithm is applied (FIGURE 4.9). In order to minimize distortion the spectral values are first normalized, after which k samples are chosen from the data (Pedregosa et al., 2011). Including more and more data results in shifting these centroids, until the most optimal centroid is found, which contains a mean coordinate for its cluster. However robust, a number of k clusters must always be chosen, and all data points within a cluster must form a convex group.

In stead of using distances between points, another spectral analysis of the dataset can be applied for clustering (FIGURE 4.10) (Pedregosa et al., 2011). Thus, the spectral coordinates of graph nodes are again treated as nodes, for which a nearest neighbour graph can be constructed. Based on normalized graph cuts, an input is created for a k-means algorithm which assigns each data point with a cluster index. Using the spectral properties of the nearest neighbour graph, the number of k clusters in automatically determined. Furthermore, this algorithm is suitable for non-convex clusters.

After clustering, nodes are filtered according to the extracted components (§ 4.2.4) and clusters (FIGURE 4.10). The clusters divide the single floor component into floor pieces, each now represented by a single node. These nodes are linked if the separated components are connected. For each floor component, the wall components in the same cluster are linked to it. The result as simplified graph $\mathcal{G}'$ is visible in FIGURE 4.11. A translation of the resulting graph back to the original model is seen in FIGURE 4.12.

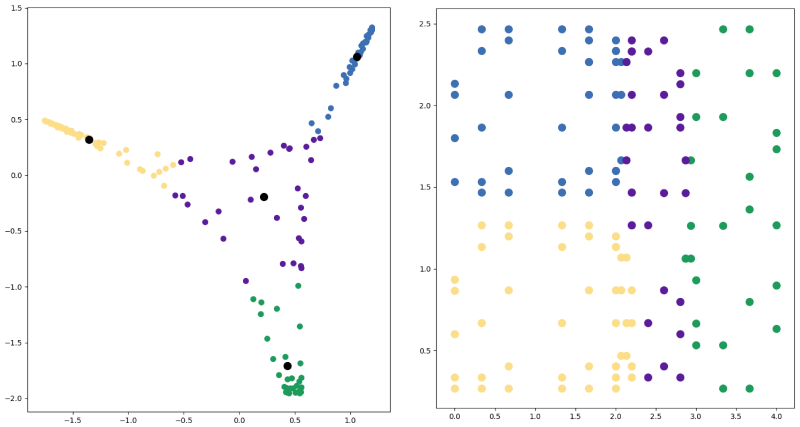


FIGURE 4.9: Results of k-means clustering of spectral values of $\mathcal{G}$ as clusters and their centres in a spectral embedding (left) and the output clusters remapped to their original positions (right).

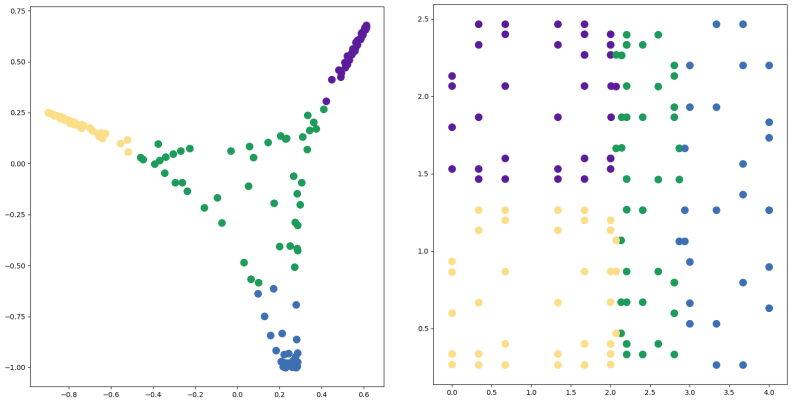


FIGURE 4.10: Results of spectral graph clustering of spectral values of $\mathcal{G}$ as clusters and their centres in a spectral embedding (left) and the output clusters remapped to their original positions (right).

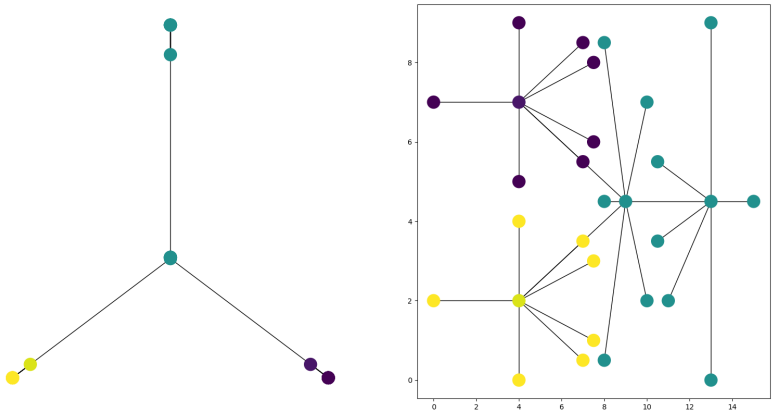


FIGURE 4.11: Resulting graph $\mathcal{G}\prime$ after clustering the spectral embedding of $\mathcal{G}$ (left in FIGURE 4.7) based on the connected components and clusters computed. The nodes remaining in the clustered graph (left) are remapped to their geometric locations (right) for visualization purposes.

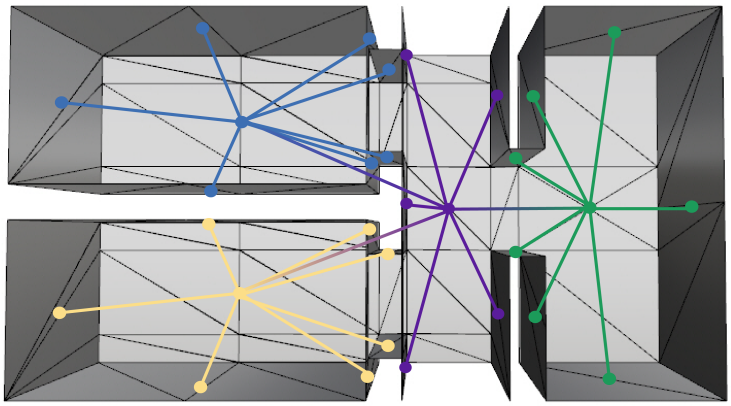


FIGURE 4.12: Conceptual visualization of the clustered graph $\mathcal{G}\prime$ representing the input mesh of the test dataset.

4.3 | Graph Extraction from BIM

The approach to extracting graph $\mathcal{H}$ from the BIM model is based on the storage and mathematical definition of the model (§ 4.1.2). The steps can be found in FIGURE 4.13.

As for mesh models, it would be possible to extract room structure from the BIM on a purely geometric basis. Then, each room would be defined by its geometric boundaries, which are the `ifcSlab` as the floor and the `ifcWall`. Walls around a room can be found as connecting or near through the natural topology of Euclidean space. The $\mathbb{R}^3$ world embedding of the origin point implies a natural topological relation between walls, specifically stored by the `RelativePlacement` element. Furthermore, it may be accompanied by a placement relative to another shape. However, a lot of coordinate computing is necessary in order to retrieve the topological relationship between building elements. Furthermore, pure geometry makes no distinction between different sides of a wall, due to which walls would become connecting entities between different rooms. Furthermore, floor elements are never stored per room. They are interpreted as structural elements, spanning across building partitions. Thus, a purely geometric approach to extracting a graph from a BIM model may be possible, though most possibly not optimal.

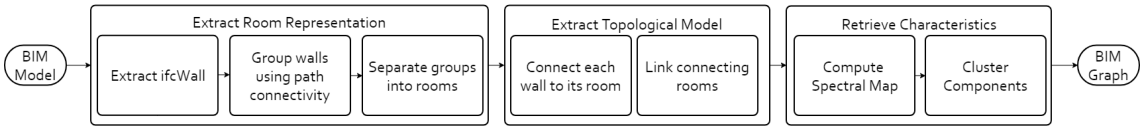


FIGURE 4.13: Method for extracting a topological graph from a BIM model.

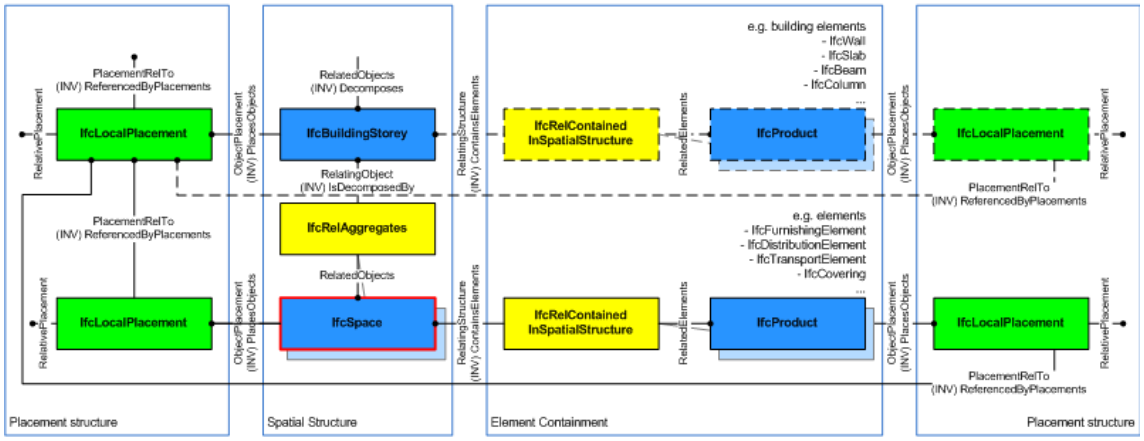


FIGURE 4.14: Composition of the ifcSpace element (BuildingSMART, 2016).

4.3.1 | Extract ifcSpace

A room in a BIM model is inherently stored as an `ifcSpace`. A space is defined as "an area or volume bounded actually or theoretically. Spaces [..] provide for certain functions within a building." (BuildingSMART, 2016). The composition of an ifcSpace is defined by its storey and placement, and defines elements within it (FIGURE 4.14). As such, it can be used to determine the distinction as well as the connection between rooms.

Then, to extract rooms from the BIM would mean to extract all `ifcSpace` entities, noting they might occur as space clusters or partitions. The relative connections between each space is defined by the attributes from the `ifcRelConnect` class. As such, a network of spaces can be built.

4.3.2 | Extract ifcWall for each ifcSpace

A wall, as a basic building element which can bound a partition of indoor space, is stored as `ifcWall`. In its standard case it is stored as a (poly)curve c built by `Points` $\mathbf{p}_i = (x, y)^T \in \mathbb{R}^2$ describing the general footprint of the element, and an extrusion depth d. Thus, an `ifcWall` is stored as a solid $\mathcal{S}(c, d)$. The coordinates are defined relative to object an origin $\mathbf{O} = (X, Y, Z)^T \in \mathbb{R}^3$ and set on a reference plane. The coordinates of the origin are relative to the full model. A full example of an `ifcWall` is listed in § C.3.

Each `ifcSpace` entity may be connected with its bounding walls by relative placement. Thus, for each `ifcSpace` a set of `ifcWall` objects may be extracted. If the same wall is related to more than one `ifcSpace`, it should be duplicated. Furthermore, the amount of `ifcDoor` elements embedded into the walls around an `ifcSpace` represent the number of links that should be laid from a single `ifcSpace` to one or more other `ifcSpace` elements.

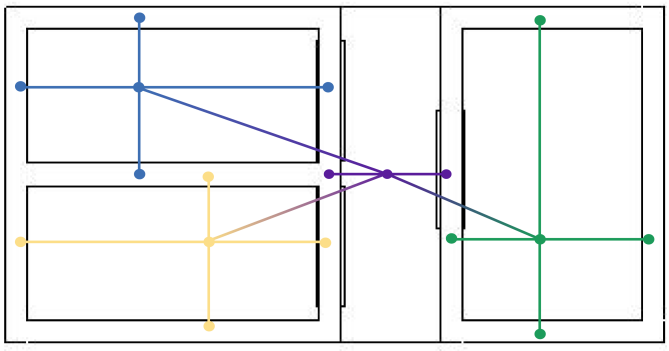


FIGURE 4.15: Conceptual visualisation of the final graph $\mathcal{H}$ extracted from the BIM model in FIGURE 4.2.

4.3.3 | Group Walls using Path Connectivity

Walls bounding the same room are related by the `Path Connectivity` element, which is used to ensure a single material or profile definition (§ C.2). Thus it can be used to group walls per room. It can be applied to check if rooms assigned to spaces in the previous step are correct and complete. If after these steps any of the walls is unassigned, the coordinates embedded in the `ifcWall` elements may be used to find their placement.

4.3.4 | Connect each Wall to a Room

A star shaped graph can be built for each room, by translating the `ifcSpace` element to a center node, surrounded by nodes representing its encapsulating `ifcWall` elements.

4.3.5 | Link Connecting Rooms

The full set of `ifcSpace` elements found within a BIM can be translated directly to a set of nodes. Links are formed by stored relationships between spaces. As a check, each `ifcDoor` between two `ifcSpace` elements should be connected to a link between nodes. Combining this structural graph with the star graphs described in § 4.3.4, a full graph representation $\mathcal{H}$ of rooms encapsulated in a BIM model can be built. A conceptual representation of the final graph is visualized in FIGURE 4.15.

4.3.6 | Compute Spectral Map

For the resulting connected graph $\mathcal{H}$, a spectral map can be computed as in § 4.2.6. The resulting spectral graphs are visualized in FIGURE 4.16.

4.4 | Data Matching

In order to enrich the mesh model created on-the-fly, it should be connected with the reference BIM. In the field of IMM attempts to match indoor spaces have been made be-

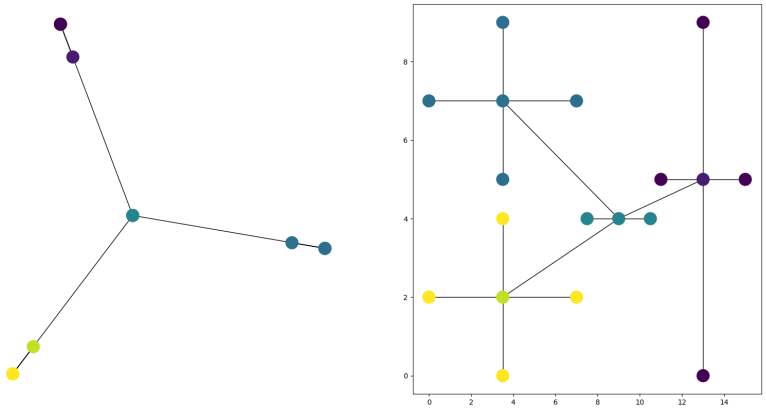


FIGURE 4.16: Visualization of spectral properties of graph $\mathcal{H}$ extracted from the BIM in FIGURE 4.2. A spectral embedding (left) compared to positional embedding (right), both coloured by its Fiedler vector.

fore, generally based on object matching. The method presented in this chapter takes a different stance, as matching is based on structure, drawn from spectral graphs.

4.4.1 | Object Matching

Generally in IMM a location is recognized as such by performing object recognition methods. Volk et al. (2014) has made an overview of object recognition methods used to build and enrich a BIM model (FIGURE 4.17). A match between the selected object type then refers to a location at which data may be retrieved or added.

For example, Huber et al. (2011) extracts a floor plan from unstructured point cloud. Tang, Huber, Akinci, Lipman, and Lytle (2010) executes instance-based object recognition by (1) generating descriptors and store them in database, (2) finding descriptors in the scene and retrieve closest matching descriptor from the database and (3) aligning objects as control step. Huber and Hebert (2003) performs automatic registration through pairwise surface matching.

4.4.2 | Spectral Graph Matching

Alternatively, the structure of a building can be used as a basis for matching, rather than separate elements within. This interpretation is used in the proposed method, where graphs as topological representation of the models are matched according to their similarity.

As mentioned in § 3.5.4, graph matching entails comparing two graphs to find whether they are (semi) isomorphic. Theoretically, two models of the same indoor structure should return two isomorphic graphs. However, as $\mathcal{G}$ and $\mathcal{H}$ are derived from two different types of models, based on different interpretations of what indoor space means, in the least the number of nodes in both models will be different (compare FIGURE 4.16 and FIGURE 4.11). As such, the graphs will not be accepted as isomorphic.

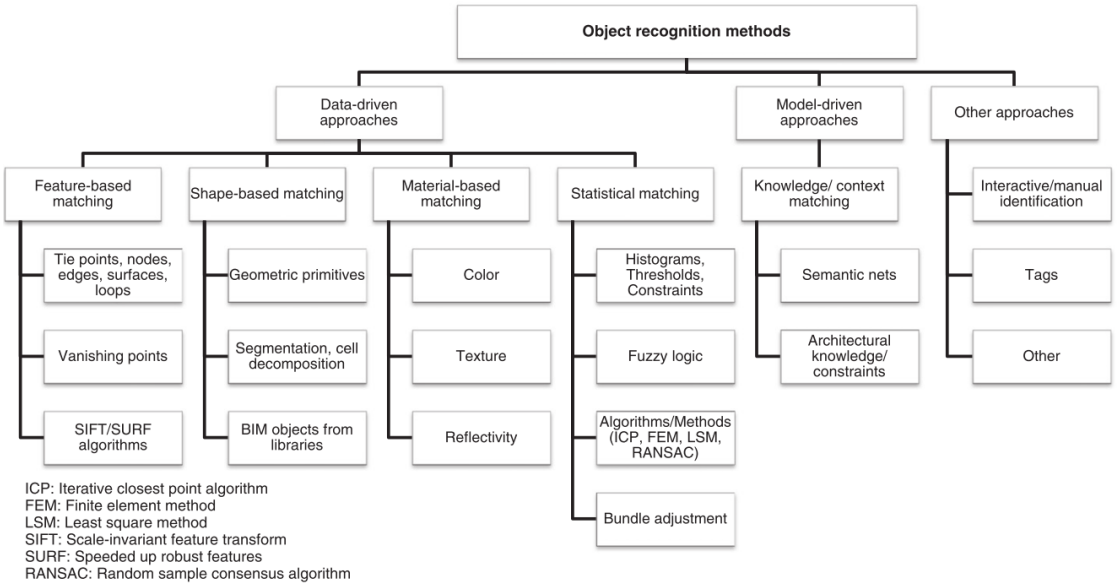


FIGURE 4.17: Object recognition approaches applied in existing buildings (Volk et al., 2014).

The option remaining would then be to construct an inexact matching process for $\mathcal{G}$ and $\mathcal{H}$. Kosinov and Caelli (2002) use spectral properties for graph matching, by embedding all nodes into a vector subspace. This method is capable of making an inexact match of co-spectral graphs, even if the number of nodes in both graphs is different. It is constructed such that the dimensionality of the graph is decreased, while minimizing loss of information. By using the matrix of eigenvectors $\mathbf{U}$, the projection of data point $\mathbf{x}$ can be constructed using EQUATION 4.3.

$$\hat{\mathbf{x}} = \mathbf{U}_k^T \mathbf{x} \tag{4.3}$$

Specifically in this research, each data point $\mathbf{x}$ represents a 2D coordinate of a node in the spectral embedding of the graph, constructed as $(\mathbf{x}_2(i), \mathbf{x}_3(i))$. The matrix of eigenvectors $\mathbf{U}$ is first constructed by decomposing the data covariance matrix $\mathbf{\Sigma}$ to its eigenvectors and eigenvalues, as in EQUATION 4.4.

$$\mathbf{\Sigma} = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^T \tag{4.4}$$

Matrix $\mathbf{U}$ used to remap coordinates $\mathbf{x}$ has a size dependent on the size and structure of the graph. To be come a useful operator, it has to be decreased to the use of the first k eigenvectors. This corresponds to the most important or influential eigenvectors. Of these first k eigenvectors, only the first few dimensions are meaningful, corresponding to the dimensionality of $\mathbf{x}$. As each $\mathbf{x}$ is a 1x2 vector, only the first two dimensions of the first $k = 2$ eigenvectors are taken.

The new embedding $\hat{\mathbf{x}}$ can be used as a basis for another clustering algorithm. If the nodes in the same cluster are similar enough to pass as a match, it is accepted. This can be

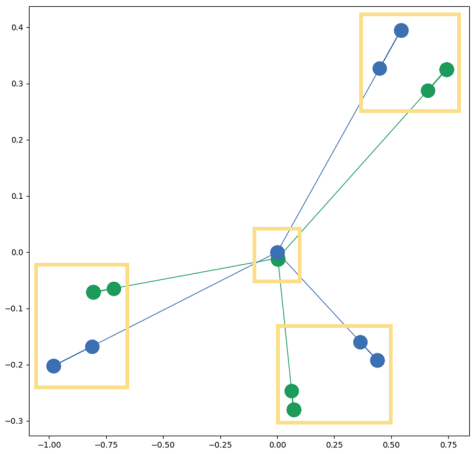


FIGURE 4.18: Clusters found in the vector subspace embedding for $\mathcal{G}$ and $\mathcal{H}$

tested based on certain attributes of the nodes, e.g.the coordinates in the original graph or the spectral values.

The match itself is based on the assumption that a graph structure implicitly stores the topology of a dataset. Thus two graphs representing the same indoor environment should be highly similar, independent of what type of model it was extracted from. Slight differences may still occur, as the interpretation of indoor space different models are based on may result in a slightly different graph. If a unique graph embedding is chosen, the structure of graphs may be compared in an intuitive manner and quite quickly a match between a scanned floor and its modelled counterpart can be made. Note that the match relies on a structure, a network of rooms, thus will be more solid as more rooms are added to the scan.

4.5 Localization Process

After a match between mesh graph $\mathcal{G}$ and reference graph $\mathcal{H}$ is found, its characteristics can be used as a basis for localization. In the mesh model, the actor capturing the indoor environment has a current position. As it is situated in the local coordinate system the mesh is stored in, this position can be connected to the closest room found in $\mathcal{G}$. The nodes of this room in $\mathcal{G}$ are connected to the same room in $\mathcal{H}$ through the matching process. These are attributed with a marker, e.g.an id, which refers to a set of semantics describing the characteristics of this room. Returning these to the actor, a location description is formed. Furthermore, the coordinate values linked to this room can provide for a basis to transform the mesh geometry such that it can be aligned with the BIM geometry. This finalization of the localization process is depicted in FIGURE 4.19.

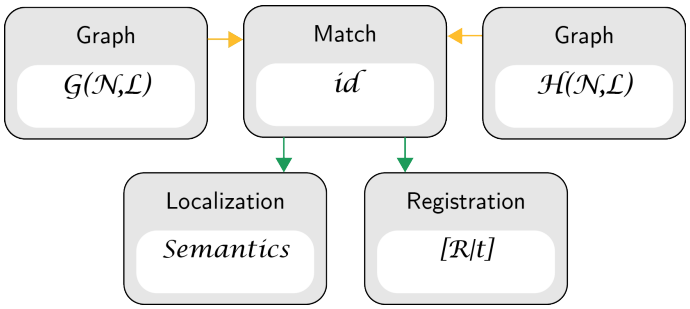


FIGURE 4.19: Method for using the match between $\mathcal{G}$ and $\mathcal{H}$ to transcend the position retrieved using SLAM on a mobile device to a location.

5 | Implementation & Results

This chapter presents experiments based upon the underlying principles of SLAM as explained in § 3.3 and describes their outcomes and conclusions to be drawn from them. Furthermore, the implementation details for the main METHODOLOGY are described, and their results are evaluated.

5.1 | Tools & Libraries

All results described in this chapter were achieved using a Tango tablet. A description of the hardware and design principles the software was based on is included in appendix x. All data was captured at the Delft University of Technology, at the faculty of Architecture & the Built Environment (BK) and the faculty of Electrical Engineering, Math and Computer Sciences (EWI). Furthermore, the code written for the methodology was tested on a series on test datasets, specifically designed to handle situations that would occur in real data.

The visualizations of the data were created in Rhino and Paraview for the meshes, and Revit for the BIM model. The code for the methodology was written in Python 2.7, using the following libraries:

- **csv** to read .obj files
- **networkx** to create, adjust and analyse graphs
- **scikit learn** to perform clustering methods
- **numpy** to perform vector and matrix calculations
- **matplotlib.pyplot** to visualize data

5.2 | Performing VI-SLAM

The aspects for performing VI-SLAM, as described in § 3.3, are examined here by performing test studies. The separate partitions described are executed simultaneously and the process is handled as the gathering of geoinformation (§ 2.2), as it models a physical location.

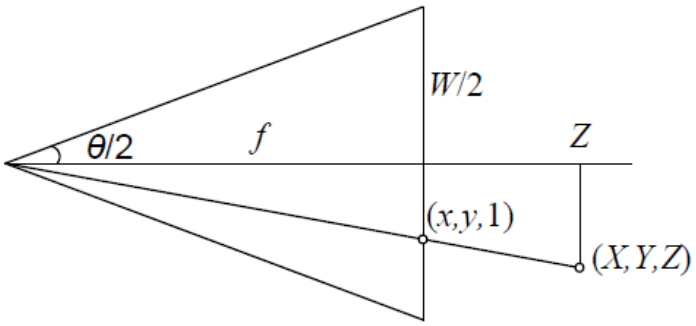


FIGURE 5.1: Depiction of central projection, to show the relationship between point coordinates $\mathbf{p} = (X, Y, X)$, image coordinates (x, y), image width W and focal length f (Szeliski, 2010).

5.2.1 | Data Capture

In order to test the capabilities of the camera used for VI-SLAM, the FOV of each of the cameras is determined. First, the intrinsic camera parameters **K** are retrieved by calibrating the device, from which the output then can be used to calculate the FOV according to EQUATION 5.1 and EQUATION 5.2. Here, W and H are the width and height of the camera image and f_x and f_y the corresponding focal lengths, corrected for the scale of each pixel. The corresponding values and calculation results can be found in TABLE 5.1. The relationship between image size and camera intrinsics is depicted in FIGURE 5.1.

$$FOV_H = 2 \cdot \arctan\left(\frac{W}{2 \cdot f_x}\right) \quad [^\circ] \tag{5.1}$$

$$FOV_V = 2 \cdot \arctan\left(\frac{H}{2 \cdot f_y}\right) \quad [^\circ] \tag{5.2}$$

Henry, Krainin, Herbst, Ren, and Fox (2010b) state that a typical camera, with a range of around 5 meters, only reaches an FOV of $\sim 60^o$. The tested RGB-D camera evaluates to average at $\sim 63^o$, and a range up to 4 meters. The fisheye camera does improve upon this with a horizontal FOV of $\sim 102^o$. However, it does not compare to the $\sim 180^\circ$ of laser scanners typically used for 3D mapping, which can provide for a much richer flow of data from each frame. Thus, the drawback in amount of data captured simultaneously has to be compensated in alignment algorithms, using extra computing power. On the other hand, a camera is capable of capturing a wider range of information that can be used for environment interpretation. Computer vision algorithms may be utilized for e.g. feature detection and colour based segmentation (Szeliski, 2010). Using the RGB-D camera as main data source thus limits the data capture as any separate camera would. However, if the broader FOV of the fisheye camera is utilized to enlarge the scene that can be captured, mobile devices equipped with both have a lead on regular cameras in capturing indoor environments.

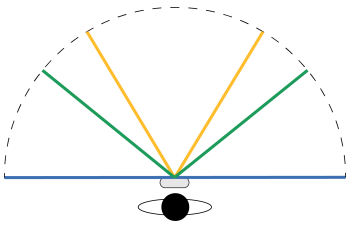


FIGURE 5.2: Calculated FOV for the RGB-D camera (yellow) and the fisheye camera (green), compared to an ideal laser scanner (blue).

5.2.2 | Data Storage

With the introduction of Tango technology, a new storage format, the Area Description File (ADF), was presented as an efficient alternative to map storage, encoding the trajectory and a map of keyframes and landmarks into binary (Google, 2017b). An interface was built around it for users to position themselves into previously scanned locations (FIGURE 5.3). However, the application was not functioning most of the time. The ADF could be written, and accessed without any visualization or further proof that an environment was actually recognized.

In another application, a polygonal mesh could be created on-the-fly (FIGURE 3.9). In addition to the geometric and topological component, colour values for each vertex are stored for visualization purposes. The mesh is stored as an indexed face-set (TABLE 4.1), allowing for fast calculation of face normals and colour values, and ensuring the orientability of the surface it represents. The specific data structure for such a mesh can be found in TABLE 5.2.

The captured data is prepared for storage by applying constant transformation to the data, as separate frames are stitched together. Each keyframe is captured relative to the camera origin, and needs to be mapped relative to the system origin, which is the camera pose at the start of the service. Thus each geometric member $\mathbf{p}_i$ of the dataset is in the least subjected to Euclidean transformation $[\mathbf{R}|\mathbf{t}]$. Most generally, rotation is stored as quaternion $\mathbf{q} = (x, y, z, w)$, which requires the set-up of rotation matrix $\mathbf{R}$, as in EQUATION 5.3.

$$\mathbf{R}_q = \begin{bmatrix} 1-2(y^2+z^2) & 2(xy-wz) & 2(xz+wy) \\ 2(xy+wz) & 1-2(x^2+z^2) & 2(yz-wx) \\ 2(wz-wy) & 2(yz+wx) & 1-2(x^2+y^2) \end{bmatrix} \tag{5.3}$$

Geometric points within a keyframe can then be connected by the creation of a mesh,

CAMERA	$W[px]$	$H[px]$	$f_x[px]$	$f_y[px]$	$FOV_H[^o]$	$FOV_V[^o]$
RGB-D	320	180	261.695	261.642	62.8831	37.9645
	640	360	523.389	523.284	62.8832	37.9645
	1280	720	1046.78	1046.57	62.8831	37.9645
Fisheye	640	480	257.952	257.727	102.255	85.9204

TABLE 5.1: Camera intrinsics influential to the FOV for the sensors embedded into the Tango tablet.

FIGURE 5.3: The interface for a Tango application that would read an ADF and position the user into a known area, for which the ADF was built previously (Google, 2017b).

which is updated by the stitching of point cloud frames. The natural topology of Euclidean space allows for meshing and updating of each keyframe, as a surface draped on the 2.5D viewshed.

5.2.3 | Data Analysis and Communication

Henry et al. (2010b) have devised a method to examine whether an RGB-D camera and the VI-SLAM algorithm executed are suitable for use in robotics, specifically for creating dense indoor maps. The experiment tests the capability of the camera to capture sufficient information for performing SLAM, including place recognition. Here, the camera was taken in large, single loops to create a map of its environment, which is then compared to a map of the examined area. The reliability of the final model is a testament to the functionality of the full process.

It is executed by scanning a large single loop, and aligning the resulting model with the ground truth. The first loop is chosen at BK, and the second at EWI, both situated at the Delft University of Technology. The tool used is the Tango Constructor, a developer tool released to display the modelling capabilities of Tango technology. A mesh is created and stored (TABLE 5.2) on-the-fly, and visualized directly in the interface.

$\mathcal{V}$							
v_i	v_X	v_Y	v_Z	R	G	B	$weight$
n_i	n_X	n_Y	n_Z				

$\mathcal{F}$			
f_i	$v_i//n_i$	$v_j//n_j$	$v_k//n_k$

TABLE 5.2: Data structure of a Tango polygonal mesh stored in a .obj format.

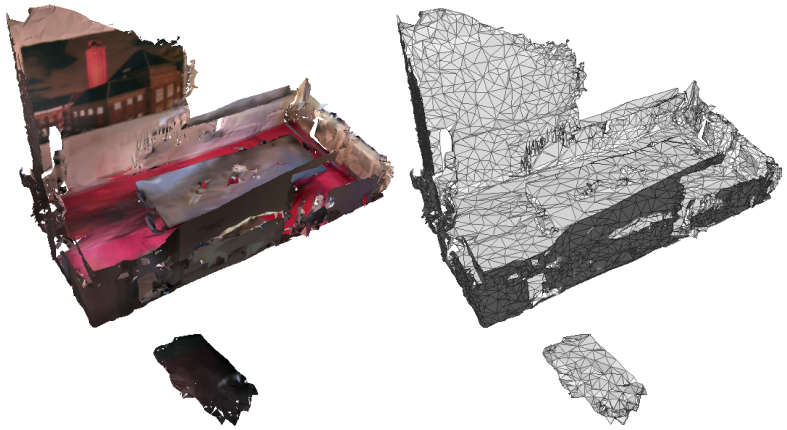


FIGURE 5.4: Example dataset of a mesh model created using VI-SLAM on a Tango device of a meeting room at the TU Delft the faculty of Architecture & the Built Environment (BK). Shown as textured surface model (left) and triangular mesh (right).

5.3 | Graph Extraction from Mesh

For the description of implementation details and results, the methodology for extracting a graph from the mesh created using VI-SLAM is executed on a dataset captured using a Tango device. The model represents a room with a large table (FIGURE 5.4), used to test correct extraction of the partially occluded floor, and handling of noise.

As described in § 4.2, the extraction of the floor from the mesh is performed by creating a graph structure, which is then filtered by a connected components algorithm testing orientation and similarity of adjacent faces. The largest connected component facing upwards is taken as floor element, all large enough components facing sideways are accepted as walls. The resulting graph is clustered according to its spectral properties into the separate rooms traversed. The result of this process is a graph of nodes representing rooms and links representing traversable connections between these rooms.

5.3.1 | Extract Topological Model

A first inquiry of data captured using the Tango device proves an apt application of gravity aware scanning, as all normal vectors $\mathcal{V} = \{v_i | n_Y > 0.5\}$ accepted as (close enough to) upwards form the horizontal surfaces in the scan (FIGURE 5.5). The histogram shown in FIGURE 5.6 furthermore confirms the notion that the largest horizontal component should represent floorspace, even when much of the area is filled with tables, as in this example. The histogram was divided into 100 bins, of which the largest returned a height of ~ -1.272, or the device was held at a height of approximately 1.25 meters while capturing the model.

A complete dual graph for the dataset in FIGURE 5.4 is depicted in FIGURE 5.7. After extracting a complete graph, the connected components algorithm is executed (ALGORITHM 1). Input are the created graph $\mathcal{G}(N, \mathcal{L})$ and the similarity threshold t_{sim}. The

FIGURE 5.5: A model created on-the-fly, coloured according to face normal direction. Green is upward as $\mathcal{V} = \{v_i | n_Y > 0.5\}$, blue to the side.

FIGURE 5.6: Side view of the mesh along with its height histogram. The largest bin corresponds to the floor set, and the second to largest to the table that is located in the middle of the room.

threshold can be entered by the actor as angle in degrees, as a measure to how much two normal vectors attributed to adjacent nodes may diverge.

5.3.2 | Extract Floors and Walls

The connected components algorithm uses a depth first search through the entire graph, to determine the similarity of neighbouring nodes. A first check is to see if the attributed normal points upwards, thus represents a horizontal surface partition, according to EQUATION 5.4. The built component is tested by size, such that only the largest floor component remains. If the first check returns a vertically oriented normal, the seed is always accepted to build a component. A second check finds the dot product similarity of the normal vectors attributed to two neighbouring nodes (n_i, n_j). With a t_{sim} set at 5 degrees, all tested datasets resulted in a graph representing floor and wall partitions only.

$$\hat{\mathbf{k}} \cdot \mathbf{n} = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \cdot \begin{pmatrix} n_x \\ n_y \\ n_z \end{pmatrix} \tag{5.4}$$

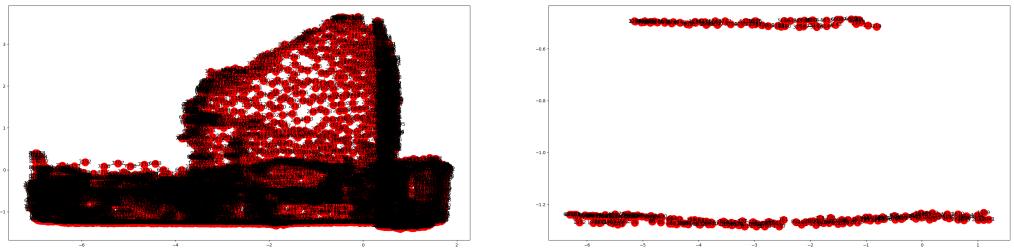


FIGURE 5.7: Complete dual graph of the model in FIGURE 5.4 (left) and the result of the connected components algorithm selecting only horizontally oriented normal vectors (right).

5.3.3 | Retrieve Characteristics

A next step is to find the bottlenecks in the graphs, or the most weakly connected partitions as a basis for creating the sparsest cut. As described in § 4.2.6, the Fiedler vector as eigenvector belonging to λ_2 is mapped to each node. A visual representation of this mapping for the test data can be found in FIGURE 4.7. The right shows a colour coded division in which the three rooms are quite well distinguished.

The sparsest cut, or the cut in the graph which requires the least or weakest connected edges to break, is found by dividing the nodes according the corresponding values in the Fiedler vector. In the simplest approach this cut is defined as median at $n_{left} < 0 < n_{right}$. A plot of these values for the test data is depicted on the left of FIGURE 4.7. This is the first cut to be found in the data, of which the result is seen on the right. Thus the first room is separated from the rest, though as expected a connecting partition of the hallway is still incorporated. A similar result is obtained by finding the next cut in the same way. However, the sparsest cut based on the Fiedler vector (FIGURE 4.8) does not return a cut at the exact border of a room. As Luxburg (2007) mentions, a clustering of a graph may be based on the sparsest cuts found in a graph. However, using a clustering algorithm on gathered spectral values optimizes the division made and returns a more natural division into more than 2 groups embedded in the graph structure.

The left image of FIGURE 4.7 shows the spectral embedding of the graph of which the coordinates are drawn as $(\mathbf{x}_2(i), \mathbf{x}_3(i))$. This embedding should return a visual representation of the underlying structure of the graph. From this graph we may conclude the traversed space is naturally divided into three main partitions and a centre. A natural clustering into four divisions as the three separate rooms and a hallway visited can be based on the gathered spectral properties.

5.3.4 | Cluster Components

The graph resulting from embedding nodes according to the graphs eigen vectors (left of FIGURE 4.7) gives a representation of the underlying structure of the indoor environment scanned. Using the coordinates of this embedding, a clustering method can be applied, in order to find a natural division into the separate rooms visited.

FIGURE 4.9 shows a clustering of the nodes using k-means. As three rooms and a hallway were visited, input for the algorithm would be $k = 4$. Giving each node an id

for the group it ends up in and remapping the nodes to their geometrical position as dual graph results in the right image. Here, it can be seen that the graph is nicely divided into four rooms. Though the results look good, a drawback to this method is the need for input, as to how many clusters should be formed.

Alternatively, spectral clustering as a method in itself is capable of estimating the number of clusters to be assigned. However, the algorithm performs best when given a number of clusters to aim at, after which the real number of clusters is estimated. With prediction and estimation set at 4 clusters, the result in FIGURE 4.12 is gained.

Comparing both results, only difference is found in the door openings between both rooms. Especially for the bottom left room, the cluster is better estimated using spectral clustering.

After performing the clustering method, each node has received a cluster id. Using the connected components formed in § 4.2.4, each cluster can be divided into sub-clusters, each representing either a floor or a wall. The distinction between both is stored in a node attribute, as normal vector direction. Now, each floor component becomes a node, which is connected to all adjacent clusters. Each wall component is connected with the floor component of the same cluster. The result is the graph in FIGURE 4.11.

5.4 Graph Extraction from BIM

As described in § 4.3, the extraction of rooms from the BIM model is based on the declaration of room defining elements. After extraction of the necessary elements, a reference graph is built.

The encoding of the source is available in a STEP-file (STEP) file, which is based on ISO 10303-21. It is a type of American Standard Code for Information Interchange (ASCII) file with a key-value structure. Each key receives an id, so that later declared values may reference to it. An example for a standard `ifcWall` declaration can be found in § C.3.

Depending on the quality of the model, some manual correction may be necessary. As the BIM is not expected to change continuously, a semi-automated generation of such a graph should be sufficient.

5.5 Data Matching

Visually comparing $\mathcal{G}$ (FIGURE 4.11) and $\mathcal{H}$ (FIGURE 4.16), the spectral embedding seems quite similar. Both graphs from a three armed star, where the first node on the arm is connected to the central node, as well as to further nodes representing the same room. However, an exact match between both models will never be made. The graphs are not purely isomorphic, if only due to the different number of nodes. However, a visual comparison does imply the possibility of an inexact match.

The difference in nodes between both models is due to the difference of interpretation of indoor space. This is most clear around the doors. A mesh model created on-the-fly would less likely contain a ceiling or the wall above the doors, as the hand-held device used to capture it would be mostly pointed downwards. The viewshed formed drapes a surface that can be divided according to the methodology described in § 4.2. Then, the

wall partitions adjoining the door would never be connected, as they are unconnected in the viewshed. On the other hand, in a BIM model, a door is only an element that can be placed into a wall. Both partitions of the wall next to it will always remain connected, as they are part of the same `ifcWall` element. Thus, it is important for the chosen matching technique to be capable of notifying graph matches between models containing a different number of nodes.

When remapping the nodes into a vector sub space, the new positions given to the nodes of $\mathcal{G}$ and $\mathcal{H}$ again are quite similar. Running a clustering algorithm results in a regrouping of nodes according to their positions in the vector subspace. Further inquiry proves a similarity between grouped nodes for both graphs, resulting in an accepted match.

In this research, only datasets representing the same number of rooms are examined. In the scale of a full BIM model, which contains more data than just the rooms traversed, a series of sub-graphs can be used for data matching. An example of such a method can be found in Vento, Cordella, Foggia, and Sansone (2004).

5.6 Localization Process

The method proposed in this research provides a possibility to perform graph-based indoor localization. However different in nature and structure, both input sources can be converted to a graph of similar calibre, such that they can be tested for a match. Before such a match can be made, a number of rooms must have been visited, in order to retrieve a meaningful graph. Thus, initializing the proposed process would entail extensive traversal through an unknown building, which would not be an ideal situation for indoor localization. Alternatively, the process could be initialized by providing an actor with a marker representing the hallway or room to start from, as well as its structure relative to the two closest partitions of indoor space to be defined.

6 Conclusions

6.1 Answer Research Questions

SQ1: What characteristics of indoor space need to be understood in order to localize an actor?

Space is defined as a mathematical structure with relational properties, to which all its members adhere (§ 2.1). As a subset of space in general, **topological space** is a structure describing the relationships between (parts of) objects that do not change under continuous deformation (§ 2.1.1). This also defines its subset **metric space**, as a set with a global distance function, where distances and angles between all of its members can be defined (§ 2.1.2). A metric space provides for a geometrical definition of its members.

Indoor Space is defined as a structure bounded by physical or functional elements, enabling human activities (§ 2.1.4). It represents a (set of) bounded functional place(s) and should be mapped in a geometrical, topological and semantic manner, in order to describe its conceptual complexity. Its elements are defined as follows:

- Geometry - What is the place the actor is in?
- Topology - What is the structure of the place the actor is in?
- Semantics - What is the meaning of the place the actor is in?

As a type of space as a structure and its subsets define in what manner embedded objects can be represented,

This structure defines in what manner embedded objects can be represented. These may be embedded into different subsets of indoor space, defined as a geometrical, topological and semantic subset. Each of these subsets may be mapped differently, either in a separate or combined model (§ 3.3.3). As such, indoor space as a data structure may be represented as in TABLE 6.1.

Localization entails positioning an actor into a meaningful context (§ 2.2.2). Thus, indoor space in which an actor can be localized must enable positioning methods, as well as provide additional contextual information about a location. Therefore, the representation of indoor space must contain geometric, topological and semantic information to fully represent its place, boundaries and function. Its place enables pure geometric positioning, its boundaries help define the underlying topological structure and its semantics help interpret its functional meaning. As such, the analysis of the topology of indoor

	GEOMETRY	TOPOLOGY	SEMANTICS
AMBIENT SPACE	Euclidean Space	Topological Space	Semantic Space
ELEMENTS	$\mathbf{P_i} = \{X, Y, Z\} \in \mathbb{E}^3$	$\mathcal{M} = \{\mathcal{V}, \mathcal{F}\} \in \mathcal{T}$	Attributes

TABLE 6.1: Characteristics of indoor space to be mapped in order to form a comprehensive model for an actor to perform indoor localization.

space allows for correct semantic assignment to its positions, so that an understanding of indoor space can be reached.

SQ2: IN WHAT WAY CAN THE SENSORS EMBEDDED IN A MOBILE DEVICE BE UTILIZED TO PERFORM INDOOR POSITIONING?

Indoor positioning not relying on a contingent system can be performed using a hybrid fusion of different sensors embedded into a mobile device (§ 3.1.2). Combining data captured by the IMU and camera, sufficient information should be retrievable, provided that the captured environment contains sufficient salient features (§ 3.2.3). Thus, indoor positioning can be performed by a VI-SLAM process, simultaneously building a geometric environment and tracking each pose and heading relatively.

Data quality may be improved by using a sensor with a wide FOV and by determining depth values for each pixel stored (§ 3.3.1). The colour and light intensity can be calculated per pixel as well.

VI-SLAM is improved by using data gathered by the IMU of the system. It defines heading, can detect sudden movement and thus eliminate the unpredictable errors this might cause. Furthermore, VI-SLAM can provide for pose change estimation in case of camera occlusion, or too little salient features.

Generally, the map built in a SLAM process consists of purely geometrical aspects. However, for IMM, a topological component is necessary as well. If the SLAM process is designed to retrieve the topological component, a polygonal mesh can be built on-the-fly. The quality of the mesh can be improved by integrating the input from all of the sensors in the device. As such, the IMU can be used for drift correction, and the camera can be used for image stitching.

Using a sufficiently enriched map, a SLAM process has been proven capable of positioning a user sufficiently accurate (§ 3.2). The reliability of both map and position can be greatly improved by integrating data from several sensors. A camera can be useful as main data source, but needs additional information to overcome commonly encountered issues. The IMU can provide for drift correction, and elaborate algorithms can provide for place recognition. The built model has to be flexible enough to adapt to these updates, in order to accurately assess a position.

In conclusion, VI-SLAM functions as an semi-autonomous positioning system on a mobile device, maximising its reliability by using multiple input sensors. The geometric model used as a basis for positioning and tracking a device can be built and updated on-the-fly, and be improved by retrieving and storing the topological structure of said model simultaneously. The final product of this positioning process then represents the geomet-

ric and topological components of the captured space. Combining the SLAM model with a reference containing semantics can transcend the retrieved set of positions to a location.

SQ3: IN WHAT WAY CAN A GRAPH MODEL OF AN INDOOR ENVIRONMENT BE ENRICHED WITH SEMANTIC INFORMATION FROM A BIM?

- A graph can be extracted from a model captured using a mobile device, as well as from a BIM
- Spectral properties of the graph model represent intrinsic structural properties of the model the graph was based on
- Similarities in spectral graphs represent similarities in structures
- The semantics of the BIM belonging to the node closest to a current position give location information

6.2 Main Conclusions

IN WHAT WAY CAN GRAPH-BASED INDOOR LOCALIZATION BE PERFORMED USING A MOBILE DEVICE, BY COMBINING SLAM AND A BIM MODEL?

- By extracting the topological structure of a captured model and a reference model
- By analysing graph properties using spectral graph theory
- By matching spectral graphs of both models, so that the current position node of one model can be enriched with the semantics of the matching node in the other

6.3 Recommendations

Though the process described in this research is not sufficient as an autonomous localization process, mainly due to the prerequisites for finding a first match as described in § 5.6, it provides for an interesting different interpretation of performing indoor localization. The graph-based matching approach could be used to enrich or improve existing methods, or provide for a basis for a new graph-based indoor localization process.

For handling the graphs used for matching, the following topics are recommended for future work:

- The place of the cut in § 4.2.7 is now set at $t = 0$. More research should go into finding the cuts of lowest conductance in stead, as the widest spread in $\mathbf{x}_2$ values may not be the most optimal place for clustering.
- Spielman (2007) states that the spectrum of the regular Laplacian $\mathbf{L}_{ij}$ used in § 4.2.6 may return fine results for regular graphs, though a safer approach would be to

extract the spectrum of the normalized Laplacian $\mathcal{L}_{ij}$, which may be defined as EQUATION 6.1 and found by operating adjacency matrix $\mathbf{A}_{ij}$ and degree matrix $\mathbf{D}_{ij}$ as $\mathbf{D}_{ij}^{-1/2}(\mathbf{D}_{ij} - \mathbf{A}_{ij})\mathbf{D}_{ij}^{-1/2}$.

- Luxburg (2007) further states that though all versions of the Laplacian matrix return similar results for regular graphs, in other cases they largely diverge. For example, using the unnormalized Laplacian only minimizes the similarity between clusters, where using a normalized version of the Laplacian can also account for similarity within a cluster. Thus, comparing results of these different approaches specifically for indoor models would be an interesting study.

- Another specification could be made by weighting the graph with similarity values. As input, several values may possibly be useful. As such, the dot product similarity of the normal vectors, the colour values of corresponding faces or a combination of both may be optional weight inputs. When using this extra input, the spectrum will be adjusted as now the adjacency matrix $\mathbf{A}_{ij}$ is defined as EQUATION 6.2 in stead of EQUATION 3.8. Luxburg (2007) states that the weights further direct placement of the sparsest cut, as greater similarity between nodes should occur within clusters. Then, a link crossing clusters will receive a lower weight and is more easily recognized as the edge of a cluster.

$$\mathcal{L}_{i,j} = \begin{cases} 1 & if \quad i = j \quad and \quad d_j \neq 0 \\ -\frac{1}{\sqrt{d_i d_j}} & if \quad (i,j) \in links \quad \mathcal{L} \\ 0 & otherwise \end{cases} \tag{6.1}$$

$$\mathbf{A}_{i,j} = \begin{cases} w_{i,j} & if \quad (i,j) \in \mathcal{L} \\ 0 & otherwise \end{cases} \tag{6.2}$$

References

Afyouni, I., Ray, C., & Claramunt, C. (2012). Spatial models for context-aware indoor navigation systems: A survey. *Journal of Spatial Information Science*, *4*(4), 85–123. Retrieved from http://josis.org/index.php/josis/article/view/73 doi: 10.5311/JOSIS.2012.4.73

Agarwal, P., Burgard, W., & Spinello, L. (2015). Metric Localization using Google Street View. *arXiv preprint arXiv:1503.04287*, 3111–3118. doi: 10.1109/IROS.2015.7353807

Armeni, I., Sener, O., Zamir, A. R., Jiang, H., Brilakis, I., Fischer, M., & Savarese, S. (2016). 3D Semantic Parsing of Large-Scale Indoor Spaces. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 1534–1543. Retrieved from http://ieeexplore.ieee.org/document/7780539/ doi: 10.1109/CVPR.2016.170

Bailey, T., & Durrant-Whyte, H. (2006). Simultaneous localization and mapping (SLAM): Part I. *IEEE Robotics and Automation Magazine*, *13*(3), 108–117. doi: 10.1109/MRA.2006.1678144

Barile, M. (2019). *Cartesian*. Retrieved 2019-01-18, from http://mathworld.wolfram.com/Cartesian.html

Borrmann, A., Amann, J., Chipman, T., Hyvärinen, J., Liebich, T., Muhic, S., ... Scarponcini, P. (2017). IFC Infra Overall Architecture Project Documentation and Guidelines. In *buildingsmart*. Retrieved from https://buildingsmart.org/wp-content/uploads/2017/07/08{_}bSI{_}OverallArchitecure{_}Guidelines{_}final.pdf

Bot, F. J., Braaksma, H. H., Braggaar, R. C., Ligtvoet, B. R., & Staats, B. R. (2016). *Using Existing Wi-Fi networks to Provide Information on Occupancy and Exploitation of Educational Facilities using at Delft University of Technology* (Tech. Rep.). Retrieved from uuid:24e94086-2c0e-4dbe-9002-446de188159b

Botsch, M., Kobbelt, L., Pauly, M., Alliez, P., & Levy, B. (2010). *Polygonal Mesh Processing* (Vol. 53). Retrieved from www.akpeters.com doi: 10.1017/CBO9781107415324.004

BuildingSMART. (2016). *openBIM*. Retrieved from https://www.buildingsmart.org/

Chung, F. (1996). *Spectral Graph Theory* (Vol. 92). Providence, Rhode Island: American Mathematical Society. Retrieved from http://www.ams.org/cbms/092http://www.math.ucsd.edu/{~}fan/research/revised.html doi: 10.1090/cbms/092

Conte, D., Foggia, P., Sansone, C., & Vento, M. (2004). Thirty Years of Graph Matching in Pattern Recognition. *International Journal of Pattern Recognition and Artificial Intelligence*, *18*(03), 265–298. Retrieved from http://www.worldscientific.com/doi/abs/10.1142/S0218001404003228 doi: 10.1142/S0218001404003228

Costin, A., Pradhananga, N., & Teizer, J. (2012). Leveraging passive RFID technology for construction resource field mobility and status monitoring in a high-rise renovation project. *Automation in Construction*, *24*, 1–15. Retrieved from http://dx.doi.org/10.1016/j.autcon.2012.02.015 doi: 10.1016/j.autcon.2012.02.015

Cromley, R. G. (1989). *Digital Cartography*. Prentice-Hall.

Dawar, A., & Khan, K. (2018, sep). Constructing Hard Examples for Graph Isomorphism. *CoRR*, *abs/1809.0*, 1–18. Retrieved from http://arxiv.org/abs/1809.08154 doi: arXiv:1809.08154v1

Diakité, A. A., & Zlatanova, S. (2016). Extraction of the 3D Free Space From Building Models for Indoor Navigation. *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences*, *IV-2/W1*(November), 241–248. Retrieved from http://www.isprs-ann-photogramm-remote-sens-spatial-inf-sci.net/IV-2-W1/241/2016/ doi: 10.5194/isprs-annals-IV-2-W1-241-2016

Díaz-Vilariño, L., Conde, B., Lagüela, S., & Lorenzo, H. (2015). Automatic detection and segmentation of columns in as-built buildings from point clouds. *Remote Sensing*, *7*(11), 15651–15667. doi: 10.3390/rs71115651

Díaz-Vilariño, L., Khoshelham, K., Martínez-Sánchez, J., & Arias, P. (2015). 3D modeling of building indoor spaces and closed doors from imagery and point clouds. *Sensors (Switzerland)*, *15*(2), 3491–3512. doi: 10.3390/s150203491

Dos Santos, D. R., Basso, M. A., Khoshelham, K., De Oliveira, E., Pavan, N. L., & Vosselman, G. (2016). Mapping Indoor Spaces by Adaptive Coarse-to-Fine Registration of RGB-D Data. *IEEE Geoscience and Remote Sensing Letters*, *13*(2), 262–266. doi: 10.1109/LGRS.2015.2508880

Eddy, M. (2015). *Google: Future Phones Will Understand, See the World*. Retrieved from http://www.pcmag.com/article2/0,2817,2485294,00.asp

Edelsbrunner, H., & Harer, J. (2010). *Computational Topology: An Introduction*. American Mathematical Society. Retrieved from http://link.springer.com/10.1007/978-3-540-33259-6{_}7 doi: 10.1007/978-3-540-33259-6_7

Ekholm, A., & Fridqvist, S. (2000). A concept of space for building classification, product modelling, and design. *Automation in construction*, *9*(3), 315–328. doi: 10.1016/S0926-5805(99)00013-8

Endres, F., Hess, J., Engelhard, N., Sturm, J., Cremers, D., & Burgard, W. (2012). An evaluation of the RGB-D SLAM system. *Icra*, *3*(c), 1691–1696. Retrieved from http://ieeexplore.ieee.org/xpls/abs{_}all.jsp?arnumber=6225199{%}5Cnpapers3://publication/doi/10.1109/ICRA.2012.6225199 doi: 10.1109/ICRA.2012.6225199

Fuentes-Pacheco, J., Ruiz-Ascencio, J., & Rendón-Mancha, J. M. (2012). Visual simultaneous localization and mapping: a survey. *Artificial Intelligence Review*, *43*(1), 55–81. doi: 10.1007/s10462-012-9365-8

Gálvez-López, D., Salas, M., Tardós, J. D., & Montiel, J. (2016, jan). Real-time monocular object SLAM. *Robotics and Autonomous Systems*, *75*, 435–449. Retrieved from http://dx.doi.org/10.1016/j.robot.2015.08.009http://linkinghub.elsevier.com/retrieve/pii/S0921889015001864 doi: 10.1016/j.robot.2015.08.009

Girard, G., Côté, S., Zlatanova, S., Barette, Y., St-Pierre, J., & van Oosterom, P. (2011). Indoor pedestrian navigation using foot-mounted IMU and portable ultrasound range sensors. *Sensors*, *11*(8), 7606–7624. doi: 10.3390/s110807606

Google. (n.d.). *Google ARCore.* Retrieved from https://developers.google.com/ar/

Google. (2017a). *Tango Customer Site.* Retrieved from https://get.google.com/tango/developers/

Google. (2017b). *Tango Developer Site.* Retrieved from https://developers.google.com/tango/

Groves, P. D. (2013). *Principles of GNSS, inertial, and multisensor integrated navigation systems* (2nd ed.). Artech House.

Henry, P., Krainin, M., Herbst, E., Ren, X., & Fox, D. (2010a). RGB-D Mapping : Using Depth Cameras for Dense 3D Modeling of Indoor Environments. *RGBD Advanced Reasoning with Depth Cameras Workshop in conjunction with RSS*, *1*(c), 9–10. Retrieved from http://ils.intel-research.net/uploads/papers/3d-mapping-iser-10-final.pdf doi: 10.1177/0278364911434148

Henry, P., Krainin, M., Herbst, E., Ren, X., & Fox, D. (2010b, apr). RGB-D mapping: Using depth cameras for dense 3D modeling of indoor environments. In *Rgbd advanced reasoning with depth cameras workshop in conjunction with rss* (Vol. 1(c)).

Hichri, N., Stefani, C., Luca, L. D., & Veron, P. (2013). Review of the « As-Built Bim » Approaches. *3D-ARCH 2013 - 3D Virtual Reconstruction and Visualization of Complex Architectures*, *XL-5/W1*(February), 107–112. doi: 10.5194/isprsarchives-XL-5-W1-107-2013

Hong, S., Jung, J., Kim, S., Cho, H., Lee, J., & Heo, J. (2015). Semi-automated approach to indoor mapping for 3D as-built building information modeling. *Computers, Environment and Urban Systems*, *51*, 34–46. Retrieved from http://dx.doi.org/10.1016/j.compenvurbsys.2015.01.005 doi: 10.1016/j.compenvurbsys.2015.01.005

Huang, A. S., Bachrach, A., Henry, P., Krainin, M., Maturana, D., Fox, D., & Roy, N. (2017). Visual Odometry and Mapping for Autonomous Flight Using an RGB-D Camera. In H. I. Christensen & O. Khatib (Eds.), *Robotics research* (Vol. 100, pp. 235 – 252). Cham: Springer International Publishing. Retrieved from http://link.springer.com/10.1007/978-3-319-29363-9 doi: 10.1007/978-3-319-29363-9

Huber, D. F., Akinci, B., Stambler, A., Xiong, X., Anil, E., & Adan, A. (2011). Methods for automatically modeling and representing as-built building information models. *In Proceedings of the NSF CMMI Research Innovation Conference*, 856558. Retrieved from https://www.ri.cmu.edu/pub{_}files/2011/1/2011-huber-cmmi-nsf-v4.pdf

Huber, D. F., & Hebert, M. (2003, jul). Fully automatic registration of multiple 3D data sets. *Image and Vision Computing*, *21*(7), 637–650. Retrieved from http://linkinghub.elsevier.com/retrieve/pii/S026288560300060X doi: 10.1016/S0262-8856(03)00060-X

Huisman, O., & de By, R. A. (2009). *Principles of GIS*. ITC Enschede.

Isikdag, U., Zlatanova, S., & Underwood, J. (2013). A BIM-Oriented Model for supporting indoor navigation requirements. *Computers, Environment and Urban Systems*, *41*, 112–123. Retrieved from http://dx.doi.org/10.1016/j.compenvurbsys.2013.05.001 doi: 10.1016/j.compenvurbsys.2013.05.001

Khoshelham, K., & Díaz-Vilariño, L. (2014). 3D modelling of interior spaces: Learning the language of indoor architecture. *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences - ISPRS Archives*, *40*(5), 321–326. doi: 10.5194/isprsarchives-XL-5-321-2014

Konolige, K. (2010). Projected texture stereo. *Proceedings - IEEE International Conference on Robotics and Automation*, 148–155. doi: 10.1109/ROBOT.2010.5509796

Kosinov, S., & Caelli, T. (2002). Inexact Multisubgraph Matching using Graph Eigenspace and Clustering Models. In T. Caelli, A. Amin, R. P. W. Duin, D. de Ridder, & M. Kamel (Eds.), *Structural, syntactic, and statistical pattern recognition* (pp. 133–142). Berlin, Heidelberg: Springer Berlin Heidelberg.

Lee, J. (2015). *Google I/O 2015 - Project Tango - Mobile 3D tracking and perception [YouTube].* Retrieved from https://www.youtube.com/watch?v=iP9m9a2KEN4

Lee, J., Li, K., Zlatanova, S., Kolbe, T. H., Nagel, C., & Becker, T. (2016). *Indoor GML.* Retrieved from http://docs.opengeospatial.org/is/14-005r4/14-005r4.html

Lemmens, M. J. P. M. (2011). *Geo-information*. Dordrecht: Springer Netherlands. Retrieved from http://link.springer.com/10.1007/978-94-007-1667-4 doi: 10.1007/978-94-007-1667-4

Lemmens, M. J. P. M. (2013). Indoor Positioning. *GIM International*(October), 5. Retrieved from https://www.gim-international.com/content/article/indoor-positioning-2

Lopez-Antequera, M., Petkov, N., & Gonzalez-Jimenez, J. (2016, oct). Image-based localization using Gaussian Processes. In *2016 international conference on indoor positioning and indoor navigation (ipin)* (pp. 1–7). IEEE. Retrieved from http://ieeexplore.ieee.org/document/7743697/ doi: 10.1109/IPIN.2016.7743697

Luxburg, U. V. (2007). A Tutorial on Spectral Clustering A Tutorial on Spectral Clustering. *Statistics and Computing*, *17*(4), 395–416. Retrieved from http://www.springerlink.com/index/10.1007/s11222-007-9033-z doi: 10.1007/s11222-007-9033-z

Marel, H. V. D. (2016). *Reference Systems for Surveying and Mapping* (Tech. Rep.). Delft University of Technology.

Matney, L. (2017). *Google retires the Tango brand as its smartphone AR ambitions move wider.* Retrieved 2018, from http://tcrn.ch/2xuH6q0

Mautz, R. (2012). *Indoor Positioning Technologies* (Vol. 85) (No. February 2012). Retrieved from http://e-collection.library.ethz.ch/eserv/eth:5659/eth-5659-01.pdf doi: 10.3929/ethz-a-007313554

Merriam-Webster. (2019). *Merriam-Webster's Dictionary.* Retrieved from https://www.merriam-webster.com/

Moteki, A., Yamaguchi, N., Karasudani, A., & Yoshitake, T. (2016). Fast and Accurate Relocalization for Keyframe-Based SLAM Using Geometric Model Selection. *IEEE Virtual Reality Conference 2016 19–23*, 235–236.

Munch, E. (2017, jul). A User's Guide to Topological Data Analysis. *Journal of Learning Analytics*, *4*(2), 47–61. doi: 10.18608/jla.2017.42.6

Nourian, P. (2016). *Configraphics: Graph theoretical methods for design and analysis of spatial configurations*. Delft: Delft University of Technology.

Nourian, P. (2018). On Topology and Topological Data Models in Geometric Modeling of

Space. (May). doi: 10.13140/RG.2.2.16572.74888

Nourian, P., Rezvani, S., Sariyildiz, I., & van der Hoeven, F. (2016). Spectral Modelling for Spatial Network Analysis. *Proceedings of the Symposium on Simulation for Architecture and Urban Design (simAUD 2016)*(May). Retrieved from https://repository.tudelft.nl/islandora/object/uuid:81c02b9c-3ddc-4273-8c2b-7e84c6dc7604

Ochmann, S., Vock, R., Wessel, R., & Klein, R. (2016). Automatic reconstruction of parametric building models from indoor point clouds. *Computers and Graphics (Pergamon)*, *54*, 94–103. Retrieved from http://dx.doi.org/10.1016/j.cag.2015.07.008 doi: 10.1016/j.cag.2015.07.008

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... Duchesnay, E. (2011). Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, *12*, 2825–2830.

Ptrucean, V., Armeni, I., Nahangi, M., Yeung, J., Brilakis, I., & Haas, C. (2015). State of research in automatic as-built modelling. *Advanced Engineering Informatics*, *29*(2), 162–171. doi: 10.1016/j.aei.2015.01.001

Ramirez, E. A. (2015). *An Experimental Study of Mobile Device Localization* (Tech. Rep.). Massachusetts Institute of Technology.

Read, R. C., & Corneil, D. G. (1977). The graph isomorphism disease. *Journal of Graph Theory*, *1*(4), 339–363. Retrieved from https://onlinelibrary.wiley.com/doi/abs/10.1002/jgt.3190010410 doi: 10.1002/jgt.3190010410

Rebolj, D., Pucko, Z., Babic, N. C., Bizjak, M., & Mongus, D. (2017). Point cloud quality requirements for Scan-vs-BIM based automated construction progress monitoring. *Automation in Construction*, *84*(October), 323–334. doi: 10.1016/j.autcon.2017.09.021

Remondino, F., & El-Hakim, S. (2006). Image-Based 3D Modelling : a Review. *The Photogrammetric Record*, *21*(September), 269–291.

Riisgaard, S., & Blas, M. R. (2004). *SLAM for Dummies* (Vol. 22) (No. June). Retrieved from http://ocw.num.edu.mn/NR/rdonlyres/Aeronautics-and-Astronautics/16-412JSpring-2005/9D8DB59F-24EC-4B75-BA7A-F0916BAB2440/0/1aslam{_}blas{_}repo.pdf doi: 10.1017/S0025315400002526

Rowland, T. (2019). *Manifold.* Retrieved 2019-01-18, from http://mathworld.wolfram.com/Manifold.html

Sattler, T., Leibe, B., & Kobbelt, L. (2012). Improving image-based localization by active correspondence search. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *7572 LNCS*(PART 1), 752–765. doi: 10.1007/978-3-642-33718-5_54

Sattler, T., Weyand, T., Leibe, B., & Kobbelt, L. (2012). Image Retrieval for Image-Based Localization Revisited. *Procedings of the British Machine Vision Conference 2012*, 76.1–76.12. Retrieved from http://www.bmva.org/bmvc/2012/BMVC/paper076/index.html doi: 10.5244/C.26.76

Schöps, T., Sattler, T., Häne, C., & Pollefeys, M. (2015). 3D Modeling on the Go: Interactive 3D Reconstruction of Large-Scale Scenes on Mobile Devices. In *Proceedings - 2015 international conference on 3d vision, 3dv 2015* (pp. 291–299). doi: 10.1109/3DV.2015

.40
Schöps, T., Sattler, T., Häne, C., & Pollefeys, M. (2016). Large-Scale Outdoor 3D Reconstruction on a Mobile Device. *Computer Vision and Image Understanding*. doi: 10.1016/j.cviu.2016.09.007
Spielman, D. A. (2007). Spectral Graph Theory and its Applications. *48th Annual IEEE Symposium on Foundations of Computer Science (FOCS'07)*(1), 29–38. Retrieved from http://ieeexplore.ieee.org/document/4389477/ doi: 10.1109/FOCS.2007.56
Stachniss, C. (2009). *Robotic Mapping and Exploration* (Vol. 55) (No. March). Berlin, Heidelberg: Springer Berlin Heidelberg. Retrieved from http://link.springer.com/10.1007/978-3-642-01097-2 doi: 10.1007/978-3-642-01097-2
Stover, C., & Weisstein, E. W. (2019). *Euclidean Space.* Retrieved 2019-01-18, from http://mathworld.wolfram.com/EuclideanSpace.html
Sweeney, C., & Turk, M. (2016). Efficient Computation of Absolute Pose for Gravity-Aware Augmented Reality Efficient Computation of Absolute Pose for Gravity-Aware Augmented. (November). doi: 10.1109/ISMAR.2015.20
Szeliski, R. (2010). *Computer Vision : Algorithms and Applications*. Springer. Retrieved from http://research.microsoft.com/en-us/um/people/szeliski/book/drafts/szelski{_}20080330am{_}draft.pdf doi: 10.1007/978-1-84882-935-0
Tang, P., Huber, D. F., Akinci, B., Lipman, R., & Lytle, A. (2010). Automatic reconstruction of as-built building information models from laser-scanned point clouds: A review of related techniques. *Automation in Construction*, *19*(7), 829–843. Retrieved from http://dx.doi.org/10.1016/j.autcon.2010.06.007 doi: 10.1016/j.autcon.2010.06.007
The Economist. (2012). *Indoor Positioning: Finding the way inside.* Retrieved from http://www.economist.com/news/technology-quarterly/21567197-navigation-technology-using-satellites-determine-your-position-only-works
Thomson, C., & Boehm, J. (2015). Automatic geometry generation from point clouds for BIM. *Remote Sensing*, *7*(9), 11753–11775. doi: 10.3390/rs70911753
Vento, M., Cordella, L. P., Foggia, P., & Sansone, C. (2004). A (sub) graph isomorphism algorithm for matching large graphs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *26*(10), 1367–1372. Retrieved from http://alpha.dmi.unict.it/dam/0607/materiale{_}didattico/VF.pdf{%}5Cnhttp://www.computer.org/portal/web/csdl/doi/10.1109/TPAMI.2004.75
Verschuren, P., & Doorewaard, H. (2010). Designing a Research Project: Project Design. *Designing a Research Project*, 1–25. Retrieved from https://www.boomlemma.nl/documenten/9789059315723{_}inkijkexemplaar.pdf doi: 10.1007/s13398-014-0173-7.2
Volk, R., Stengel, J., & Schultmann, F. (2014). Building Information Modeling (BIM) for existing buildings - Literature review and future needs. *Automation in Construction*, *38*, 109–127. Retrieved from http://dx.doi.org/10.1016/j.autcon.2013.10.023 doi: 10.1016/j.autcon.2013.10.023
Weisstein, E. W. (n.d.-a). *Isomorphic.* Retrieved 2019-01-18, from http://mathworld.wolfram.com/Isomorphic.html

Weisstein, E. W. (n.d.-b). *Isomorphic Graphs.* Retrieved 2019-01-18, from http://mathworld.wolfram.com/IsomorphicGraphs.html

Weisstein, E. W. (2019a). *Dual Graph.* Retrieved 2019-01-18, from http://mathworld.wolfram.com/DualGraph.html

Weisstein, E. W. (2019b). *Metric Space.* Retrieved 2019-01-18, from http://mathworld.wolfram.com/MetricSpace.html

Weisstein, E. W. (2019c). *Topology.* Retrieved 2019-01-18, from http://mathworld.wolfram.com/Topology.html

Williams, B., Cummins, M., Neira, J., Newman, P., Reid, I., & Tardós, J. (2009). A comparison of loop closing techniques in monocular SLAM. *Robotics and Autonomous Systems*, *57*(12), 1188–1197. Retrieved from http://dx.doi.org/10.1016/j.robot.2009.06.010 doi: 10.1016/j.robot.2009.06.010

Wolfram | Alpha. (2019a). *Ambient.* Retrieved 2019-01-18, from https://www.wolframalpha.com/input/?i=ambient

Wolfram | Alpha. (2019b). *Space.* Retrieved 2019-01-18, from https://www.wolframalpha.com/input/?i=space

Worboys, M. F. (2011). Modeling indoor space. *Proceedings of the 3rd ACM SIGSPATIAL International Workshop on Indoor Spatial Awareness - ISA '11*, 1–6. Retrieved from http://dl.acm.org/citation.cfm?doid=2077357.2077358 doi: 10.1145/2077357.2077358

Worboys, M. F., & Duckham, M. (2004). *GIS: A Computing Perspective*. doi: 10.1017/S0016756800007718

Xiao, J., Zhou, Z., Yi, Y., & Ni, L. M. (2016). A Survey on Wireless Indoor Localization from the Device Perspective. *ACM Computing Surveys*, *49*(2), 1–31. Retrieved from http://dl.acm.org/citation.cfm?doid=2966278.2933232 doi: 10.1145/2933232

Xiong, X., Adan, A., Akinci, B., & Huber, D. (2013). Automatic creation of semantically rich 3D building models from laser scanner data. *Automation in Construction*, *31*, 325–337. Retrieved from http://dx.doi.org/10.1016/j.autcon.2012.10.006 doi: 10.1016/j.autcon.2012.10.006

Yassin, A., Nasser, Y., Awad, M., Al-Dubai, A., Liu, R., Yuen, C., & Raulefs, R. (2016). Recent Advances in Indoor Localization: A Survey on Theoretical Approaches and Applications. *IEEE Communications Surveys Tutorials*, *PP*(99), 1. doi: 10.1109/COMST.2016.2632427

Zafari, F., Gkelias, A., & Leung, K. (2017). A Survey of Indoor Localization Systems and Technologies. , 1–30. Retrieved from http://arxiv.org/abs/1709.01015 doi: 10.1109/SIU.2014.6830467

Zeibak-Shini, R., Sacks, R., Ma, L., & Filin, S. (2016). Towards generation of as-damaged BIM models using laser-scanning and as-built BIM: First estimate of as-damaged locations of reinforced concrete frame members in masonry infill structures. *Advanced Engineering Informatics*, *30*(3), 312–326. Retrieved from http://dx.doi.org/10.1016/j.aei.2016.04.001 doi: 10.1016/j.aei.2016.04.001

Zhou, Y., Ding, L., Wang, X., Truijens, M., & Luo, H. (2015). Applicability of 4D modeling for resource allocation in mega liquefied natural gas plant construction. *Automa-*

tion in Construction, *50*(C), 50–63. Retrieved from http://dx.doi.org/10.1016/j.autcon.2014.10.016 doi: 10.1016/j.autcon.2014.10.016

Zlatanova, S., Liu, L., Sithole, G., Zhao, J., & Mortari, F. (2014). *Space subdivision for indoor applications* (No. 66).

Zlatanova, S., Sithole, G., Nakagawa, M., & Zhu, Q. (2013). Problems in indoor mapping and modelling. *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences - ISPRS Archives*, *40*(4W4), 63–68. doi: 10.5194/isprsarchives-XL-4-W4-63-2013

A | The Tango Platform

The Tango platform is comprised of a software stack made available onto Tango enabled devices. The stack contains Application Programming Interfaces (APIs) and Software Development Kits (SDKs) for developers to build upon. The device itself has to run on Tango core technology and contains the sensors deemed necessary for its functionality.

The underlying principles of Tango technology are based on computer vision, and are presented as threefold (Google, 2017b; Lee, 2015): *motion tracking*, *area learning* and *depth perception* (FIGURE A.1). These concepts are designed to have a device interpret its surroundings as close to human perception as possible. Therefore, the technology underlying these principles may intertwine at all times, but the culmination of the technologies is expected to accumulate understanding of space, as humans would unconsciously.

A.1 | Motion Tracking

Motion tracking can be defined as the registration of *movement* the device undergoes in a *session*, relative to its *starting point*.

Movement is recorded as the pose of the device in Six Degrees of Freedom (6DoF), defined in pitch, roll and yaw. This egomotion is captured by the IMU and stored as a 3D translation vector $\mathbf{t} = (t_X, t_Y, t_Z)^T$ and a rotation quaternion $\mathbf{q} = (x, y, z, w)$. This **pose** is constantly updated, using pose estimation and correction algorithms. Furthermore, the wide FOV of the fisheye camera is utilized to detect landmarks, and track their rel-

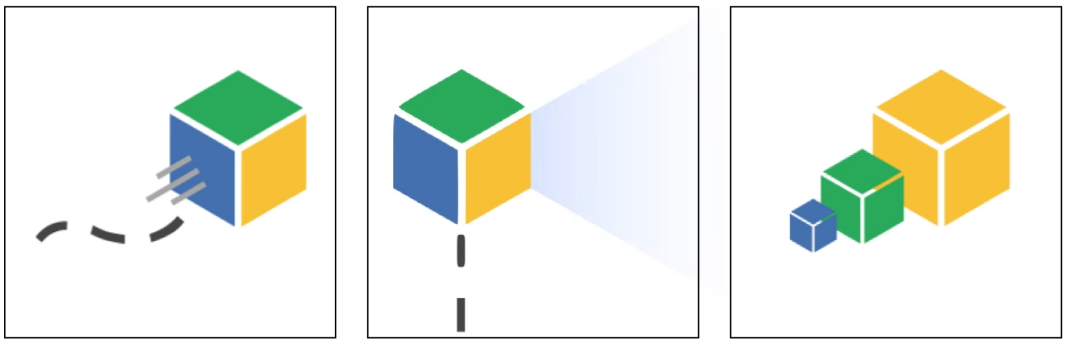


FIGURE A.1: Basic concepts underlying Tango technology. (Google, 2017b)

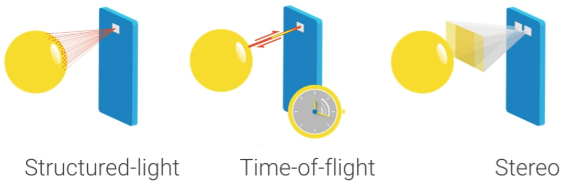


FIGURE A.2: Depth perception methods included into Tango technology. (Google, 2017b)

ative movement across frames. These data sources are combined in real time, tracking movement at 100 updates per second.

The duration of a *session* is defined by the application. At the start of a session, the current pose of the device, now the *starting position*, is stored as point of **origin**, and a 3D reference frame is built around it. Further movement is tracked relatively, and can be plotted onto a 2D grid. When a motion tracking session is finished, its data is not stored.

A.2 | Area Learning

For Tango technology, area learning is defined as the registration of a *location*, so that it can be *recognized* by a positioning alogrithm.

Location representing instances are defined as landmarks and keyframes. These are stored inside the ADF, and encoded to binary. The ADF can be read by the accompanied application only, which would then perform the place recognition process.

A location is *recognized* when the camera detects a sufficient amount of landmarks similar enough to ones stored in the ADF. The current pose is then stored as session origin, accompanied by the relative difference to the ADF origin. Thus, two coordinate systems are defined upon $\mathbb{E}^3$ space, as session $S(\mathbf{t}_S, \mathbf{q}_S)$ and ADF $A(\mathbf{t}_A, \mathbf{q}_A)$. The scale of both systems is expected to be the same. Further movement is mirrored to the ADF, and new pose estimation is improved by performing drift correction (§ 3.3.2).

A.3 | Depth Perception

Depth perception is accomplished through the *estimation of distance* to objects, to increase accuracy and precision of both motion tracking and area learning. Here, three different possible methods are utilized (FIGURE A.2):

The devices' active IR sensor is used to perform structured-light depth perception, by calculating the area of the projected IR beam onto a single object. The same data can be used to calculate time-of-flight, as the time it takes an IR beam to travel to an object, and back to the sensor. The stereo system utilizes imagery from both camera's to calculate depth through photogrammatric triangulation. These different depth measurements can be interpolated, then stitched to create a frame-by-frame point cloud (FIGURE 3.5).

A.4 | Tango Enabled Devices

Tango technology was introduced in 2014 on the Peanut phone, a pilot that was discontinued in September 2015 (Google, 2017b). The next release was on the Tango Development kit, which contained the Yellowstone tablet. The Lenovo Phab 2 Pro and the ASUS ZenFone AR have been released as commercially available Tango enabled devices.

This research has been performed using the Tango development kit. The included tablet has a 7.02" 1920x1200 HD IPS display (323 ppi) and runs on Android 4.4 KitKat. It runs on a NVIDIA Tegra K1 processor and has 128 GB flash memory and 4 GB RAM. The sensors embedded into the device and their function in SLAM processes can be found in TABLE A.1. Additionally, a GNSS receiver is embedded, aided by a compass and barometer for a faster and more precise outdoor location fix.

The new combination of sensors into a hand-held device should enable a plethora of new possible applications interesting to the consumer. Therefore, Tango technology was introduced accommodated by an open call for developers, to create new applications on the platform, using its unique benefits. In order to create this possibility for developers, a website was published (Google, 2017b), containing descriptions of the technology, as well as the software stack developers can build their applications on. However, with the discontinuation of the project, support and documentation became unavailable.

SENSOR	FUNCTIONS
4MP 2 μm RGB-IR Camera	Generate Stereo Images
	Generate Point Cloud (Passive)
Fisheye Camera	Generate Stereo Images
	Motion Tracking
IR projector	Generate Point Cloud (Active)
IMU: Accelerometer	Acceleration Tracking
IMU: Gyroscope	Pose Estimation

TABLE A.1: Sensors embedded into the Tango tablet and their functionality for indoor localization and modelling.

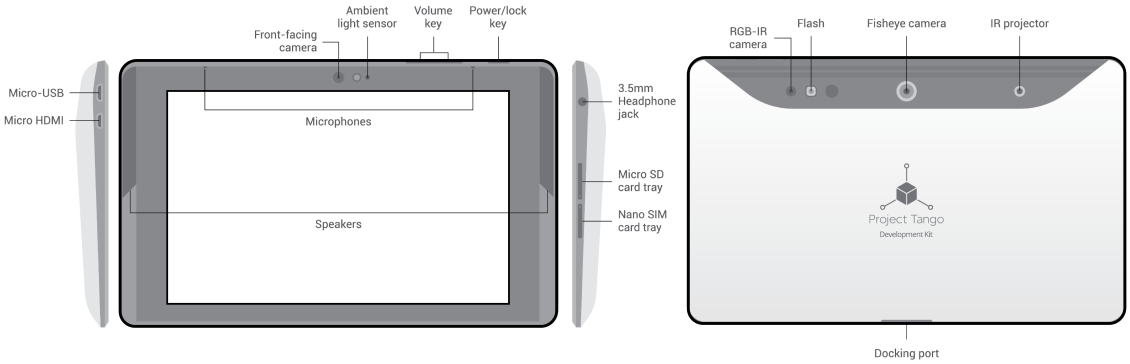


FIGURE A.3: Hardware diagram for the Tango tablet. (Google, 2017b)

A.5 | Research using the Tango Device

Since the release of Tango technology, the device has been used in several research projects for its specific sensor combination in a hand-held device. It has mostly been used as a tool for examining theories and algorithms.

The Tango tablet's capabilities in 3D modelling for the purpose of indoor localization was tested by Diakité and Zlatanova (2016), aiming at examining the extents of Tango technology itself. The Tango tablet's 3D modelling capabilities in large-scale outdoor scenes have been tested by Schöps, Sattler, Häne, and Pollefeys (2015), using monocular motion stereo to reconstruct scenes on the fly. This is based on the device's VIO and images generated from the fish eye camera by first computing frame by frame depth maps, and then fusing them using volumetric depth map fusion. To improve the accuracy of the model several filtering steps are performed, which are improved in a later set-up (Schöps, Sattler, Häne, & Pollefeys, 2016). Furthermore, the applicability to other devices is discussed, rendering the Tango tablet to merely a useful tool for the development of this method.

In another outdoor application, Agarwal, Burgard, and Spinello (2015), have combined the device's VIO with geotagged images from Google Streetview, in order to obtain accurate metric localization. Then, the Tango tablet is used to implement and evaluate the developed system for the use of personal localization in an urban scenario. However, when performing image matching from such large databases, geometric bursts will occur, a problem to which Sattler, Leibe, and Kobbelt (2012) have attempted to find a solution using the Tango tablet. Both research projects handle localization as an image matching problem.

Ramirez (2015) has used the Tango device to generate data and implement an application for the evaluation of text-based SLAM algorithms. Sweeney and Turk (2016) developed an AR application for the device to localize itself into a reference 3D point cloud using its absolute position. This is determined through gravity calculations performed using the devices IMU, a technique used to improve the accuracy of imaged based SLAM.

B | Loop Experiment

In order to test an RGB-D camera and SLAM algorithm for performing VI-SLAM, Henry et al. (2010b) scan a large closed loop and aligning the results with the ground truth. The execution of loop closure is visualized in active updates of the ouput mesh, as the user collects more data. When a loop is finished, the user returns to an environment that is already scanned. The combination of many data sources in the used device should account for most of the drift occurring during the scan, thus objects that have been captured before should be found in a position it has been placed at. If not, according to SLAM principles, this place should be recognized, and the tracked pose of the user should be aligned with the stored pose according to the captured features, as a sign of adaptive positioning. Thus, closed loop capturing using apt VI-SLAM execution allows for a solid representation of an environment, as the place recognition process at the end of the loop accounts for drift that has occurred.

B.1 | Loop 1: BK

The loop at BK was focussed on capturing the inside bounds of the route, to match with the floor plan. In this environment, the walls contained too many, and the floors too little salient features. Considering this tough challenge, the result is still quite good (FIGURE B.2). The route began at the bottom left of the image, and continued following the brightly coloured carpet. Before the first corner the lack of useful features in both floor and wall proposed problems for the device. While attempting to capture the model, the current pose was suddenly repositioned a few meters back, due to the lack of change in the environment; a faulty execution of place recognition. In the model further examined, this did not happen. In the top right corner a highly reflective floor is reached, and after turn the route is bordered by a full glass wall. Due to these environmental challenges the quality of the scan largely deteriorated.

The distortion in the loop at BK effectuates in a model displacement visible at the end of the loop (FIGURE B.3). However, the displacement is mainly linear, and the model is only slightly affected on angle. This implies that orientation changes may be accounted for in the process by the IMU. However, the lack of features along the way can easily result in recorded paths containing slight length distortions, immediately resulting in an unclosed loop. Considering the small amount of landmarks the device could detect while tracing the reflective and repetitive surroundings, the finalized loop turns out quite okay, but not reliable enough for precise positioning.

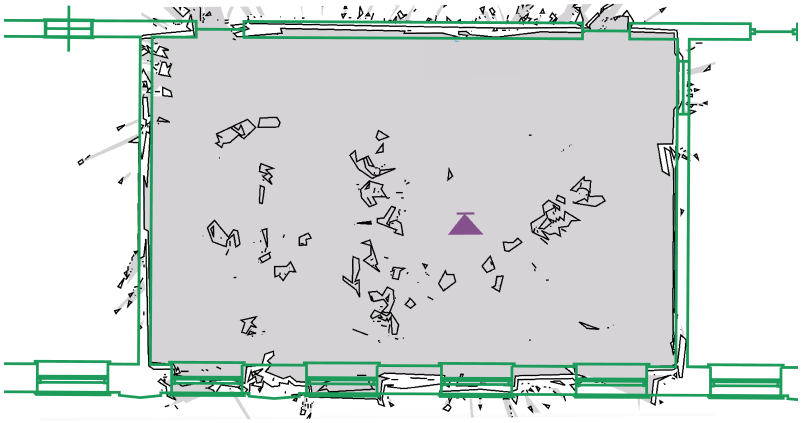


FIGURE B.1: Floorplan created using VI-SLAM compared to the ground truth. The used algorithm classifies elements with a limited height as furniture, which is then deleted from the map.

Furthermore, in another recording of the same loop, it was executed until the technology at least attempted to connect the model of the starting and ending point. However, only the faces belonging to the couch used as marker for this start and finish were recoloured, according to the newly recorded data. The model itself remained as originally captured, not adapted by the drift correction. This may be due to the fact that capture and visualization are performed simultaneously.

B.2 | Loop 2: EWI

The loop at EWI was focused on capturing as much of the floor as possible, while including a small part of the wall. It contains a height difference in two sets of stairs, and patterned floors which contains many visual features as possible landmarks.

The route was started at the white table, which can be seen at the top of FIGURE B.4, then moves to the left of the image. At the start of the loop an error occurred due to overexposure to light, even though the scene was scanned so that as many features as possible would still be visible. Even though the scan was initiated facing away from the sun, the light from the windows onto reflective surfaces directly deteriorated the quality of the scan. The resulting lapse has immediate consequences for the quality of the model, as direct horizontal displacement followed (FIGURE B.4), which persisted until the end of the loop (FIGURE B.5). At the end of the route, the stairs taken down were not captured by the cameras, resulting in a vertical displacement as well (FIGURE B.6). The rest of the model is perfectly aligned on the Z-axis.

In both cases, a route correction based on the device's IMU could have been applied. This combination of input from several sensors would improve the quality of the captured data (§ 3.2). The dependence of the algorithm on visual input implies that sensor integration was not applied here, though more input would have been available.

B.3 Conclusion

According to the execution of this experiment, the tool used does not execute the loop closure process sufficiently in the scanning of a large loop. However, on a smaller scale, the results are much reliable (FIGURE B.1). The difference could arise from the strain on the processor and limitations to the executed algorithms in capturing the large loops, as each keyframe update is directly visualized.

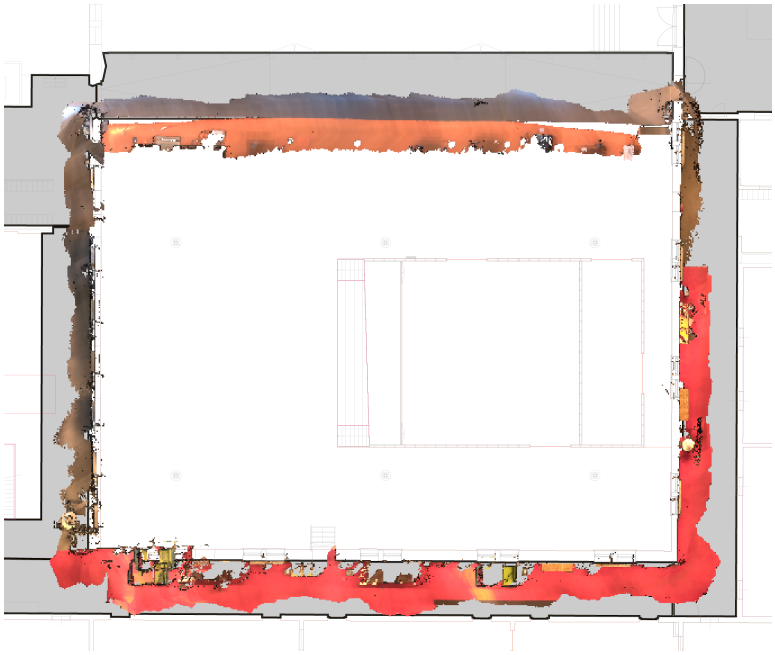


FIGURE B.2: Modelled loop at BK projected onto floor plan.

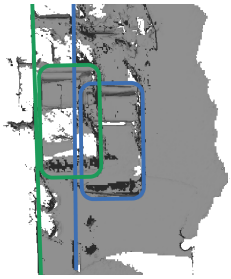


FIGURE B.3: Horizontal displacement of the model in FIGURE B.2. Indicated by couch and wall for start of loop (blue) and end of loop (green).

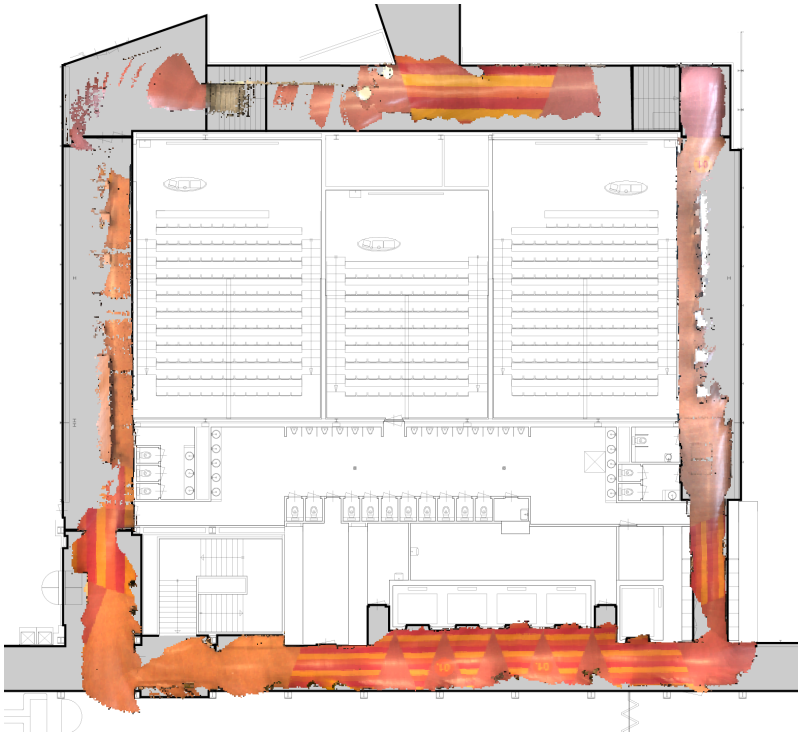


FIGURE B.4: Modelled loop at EWI projected onto floor plan.

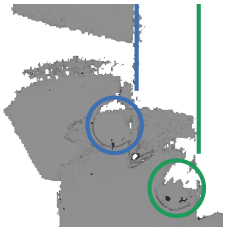


FIGURE B.5: Horizontal displacement of the model in FIGURE B.4. Indicated by table and staircase handle for start of loop (blue) and end of loop (green).

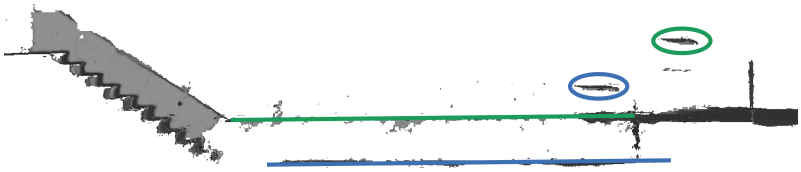


FIGURE B.6: Vertical displacement of the model in FIGURE B.4. Indicated by table and floor height for start of loop (blue) and end of loop (green).

C | ifc Data Schema's

This appendix shows the data schema's for relevant BIM components, as mentioned in § 4.3. It includes the definition of types of an `ifcBuildingElement`, the schema and an example for an `ifcWall` Standard Case and the path connectivity relational schema for material definition of building elements, such as an `ifcWall`.

C.1 | ifcBuildingElement

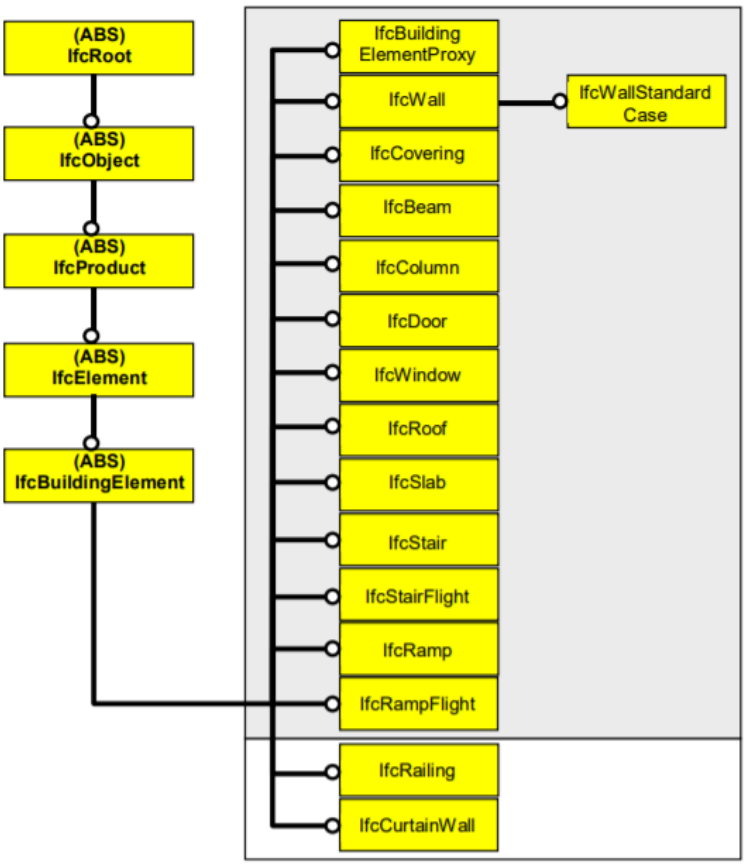


FIGURE C.1: Declaration for types of `ifcBuildingElement`.

C.2 | Path Connectivity

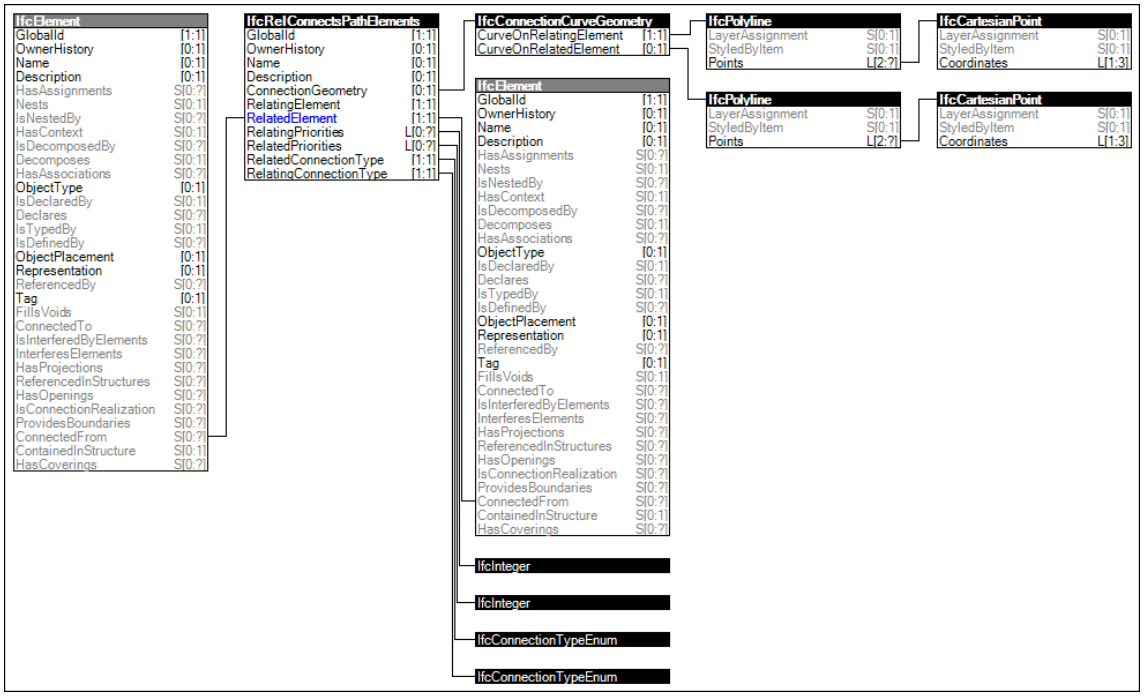


FIGURE C.2: Schema for path connectivity as a relationship indicating parameter for material layers or profiles. http://www.buildingsmart-tech.org/ifc/IFC4/Add2/html/schema/templates/diagrams/path-connectivity.png

C.3 | ifcWall

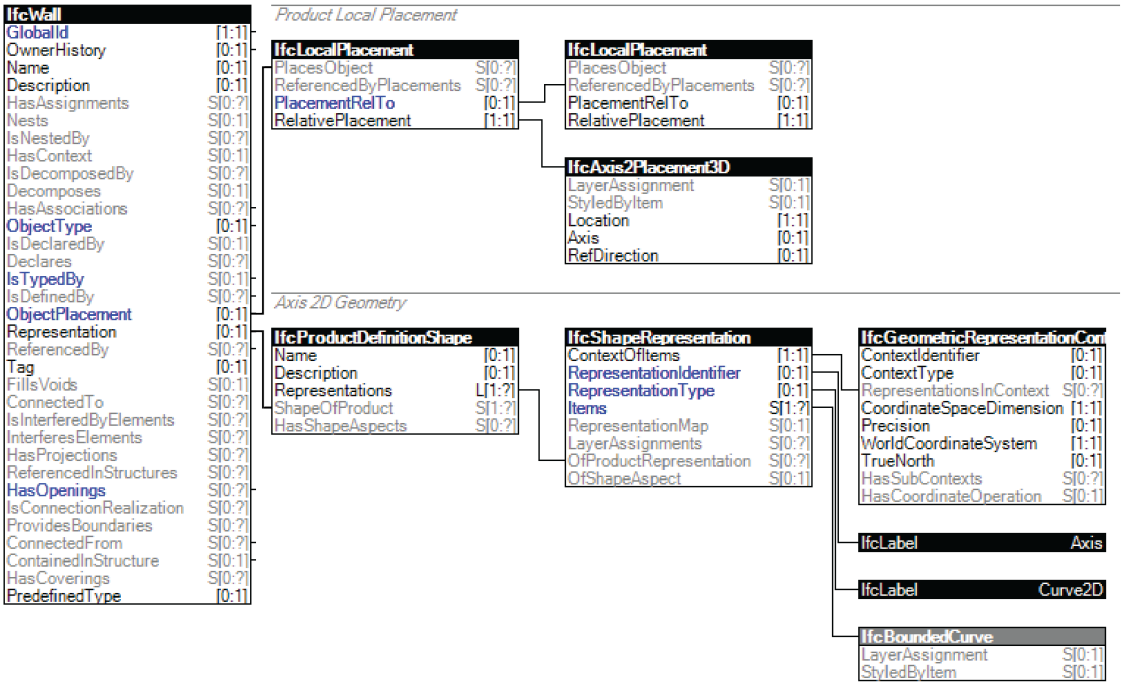


FIGURE C.3: Part of the schema for an `ifcWall` building element, containing the placement of its geometry, as well as an example for possible geometry. Full schema at http://www.buildingsmart-tech.org/ifc/IFC4/Add2/html/diagrams/general-usage/ifcwall.png.

C.4 | ifcWall Standard Case

```
ISO-10303-21;
HEADER;
FILE_DESCRIPTION(('ViewDefinition [DesignTransferView_V1]'),'2;1');
/* name */
/* time_stamp */
/* author */
/* organization */
/* preprocessor_version */
/* originating_system */
/* authorization */
FILE_NAME('','2016-02-04T08:47:55',('Jon'),('Unknown'),
'GeomGymIFC by Geometry Gym Pty Ltd','Unknown Application','None');
FILE_SCHEMA(('IFC4'));
ENDSEC;
DATA;
/* general entities required for all IFC data sets, defining the
 context for the exchange */
#1=IFCGEOMETRICREPRESENTATIONCONTEXT($,'Model',3,0.0001,#3,$);
#2=IFCCARTESIANPOINT((0.0,0.0,0.0));
#3=IFCAXIS2PLACEMENT3D(#2,$,$);
#4=IFCGEOMETRICREPRESENTATIONSUBCONTEXT('Axis','Model',*,*,*,#1,$,
.MODEL_VIEW.,$);
#5=IFCGEOMETRICREPRESENTATIONSUBCONTEXT('Body','Model',*,*,*,#1,$,
.MODEL_VIEW.,$);
/* defines the default building (as required as the minimum spatial
 element) */
#50=IFCBUILDING('39t4Pu3nTC4ekXYRIHJB9W',#56,'IfcBuilding',
 $,$,$,$,$,$,$,$,$,$);
#51=IFCPERSONANDORGANIZATION(#52,#53,$);
#52=IFCPERSON('Jon','Jon',$,$,$,$,$,$);
#53=IFCORGANIZATION($,'Geometry Gym Pty Ltd',$,$,$);
#54=IFCAPPLICATION(#55,'0.0.1.0','ggRhinoIFC - Geometry Gym Plug-in for
 Rhino3d','ggRhinoIFC');
#55=IFCORGANIZATION($,'Geometry Gym Pty Ltd',$,$,$);
#56=IFCOWNERHISTORY(#51,#54,$,.ADDED.,1454575675,$,$,1454575675);
#57=IFCRELCONTAINEDINSPATIALSTRUCTURE('3Sa3dTJGn0H8TQIGiuGQd5',#56,
 'Building','Building Container for Elements',(#303),#50);
#58=IFCAXIS2PLACEMENT3D(#2,$,$);
```

```
#100=IFCPROJECT('0$WU4A9R19$vKWO$AdOnKA',#56,'IfcProject',$,$,$,$,(#1),#101);
#101=IFCUNITASSIGNMENT((#102,#103,#104));
#102=IFCSIUNIT(*,.LENGTHUNIT.,.MILLI.,.METRE.);
#103=IFCSIUNIT(*,.PLANEANGLEUNIT.,$,.RADIAN.);
#104=IFCSIUNIT(*,.TIMEUNIT.,$,.SECOND.);
#105=IFCRELAGGREGATES('091a6ewbvCMQ2Vyiqspa7a',#56,'Project Container',
'Project Container for Buildings',#100,(#50));
#200=IFCMATERIAL('Masonry - Brick - Brown',$,$);
#202=IFCMATERIAL('Masonry',$,$);
#204=IFCMATERIALLAYER(#200,110.0,.U.,'Finish',$,$,$);
#206=IFCMATERIALLAYER($,50.0,.T.,'Air Infiltration Barrier',$,$,$);
#208=IFCMATERIALLAYER(#202,110.0,.U.,'Core',$,$,$);
#210=IFCMATERIALLAYERSET((#204,#206,#208),'Double Brick - 270',$);
#211=IFCRELASSOCIATESMATERIAL('36U74BIPDD89cYkx9bkV$Y',#56,'MatAssoc',
 'Material Associates',(#300),#210);
#300=IFCWALLTYPE('2aG1gZj7PD2PztLOx2$IVX',#56,'Double Brick - 270',
 $,$,$,$,$,$,.NOTDEFINED.);
#301=IFCRELDEFINESBYTYPE('1$EkFElNT8TB_VUVG1FtMe',#56,$,$,(#303),#300);
#302=IFCRELDECLARES('1lEof85zvB$O57GEVffll1',#56,$,$,#100,(#300));
#303=IFCWALL('0DWgwt6o1FOx7466fPk$jl',#56,$,$,$,#306,#318,$,$);
#304=IFCMATERIALLAYERSETUSAGE(#210,.AXIS2.,.POSITIVE.,0.0,$);
#305=IFCRELASSOCIATESMATERIAL('1BYoVhjtLADPUZYzipA826',#56,'MatAssoc',
 'Material Associates',(#303),#304);
#306=IFCLOCALPLACEMENT($,#307);
#307=IFCAXIS2PLACEMENT3D(#2,$,$);
#308=IFCCARTESIANPOINT((5000.0,0.0));
#309=IFCCARTESIANPOINT((0.0,0.0));
#310=IFCPOLYLINE((#309,#308));
#311=IFCSHAPEREPRESENTATION(#5,'Axis','Curve2D',(#310));
#312=IFCRECTANGLEPROFILEDEF(.AREA.,'Wall Perim',#313,5000.0,270.0);
#313=IFCAXIS2PLACEMENT2D(#314,$);
#314=IFCCARTESIANPOINT((2500.0,135.0));
#315=IFCDIRECTION((0.0,0.0,1.0));
#316=IFCEXTRUDEDAREASOLID(#312,$,#315,2000.0);
#317=IFCSHAPEREPRESENTATION(#5,'Body','SweptSolid',(#316));
#318=IFCPRODUCTDEFINITIONSHAPE($,$,(#311,#317));
ENDSEC;
END-ISO-10303-21;
```

FIGURE C.4: Source: http://www.buildingsmart-tech.org/ifc/IFC4/Add2/html/annex/annex-e/wall-standard-case.ifc.htm

Colophon

This document was typeset using LaTeX and compiled using TeXStudio.

The figures and diagrams were drawn using Adobe Illustrator CC 2015.

TU Delft