Connecting Semantic Building Information Models and Robotics: An application to 2D LiDAR-based localization

R.W.M. Hendrikx\textsuperscript{1}, P. Pauwels\textsuperscript{1}, E. Torta\textsuperscript{1}, H.P.J. Bruyninckx\textsuperscript{1} and M.J.G. van de Molengraft\textsuperscript{1}

Abstract—Much of the state of the art localization approaches require a geometric map that is created using Simultaneous Localization and Mapping (SLAM) methods. The creation, annotation and maintenance of such a map requires manual labor and expertise which can have an impact on efficient robot deployment, particularly in large buildings. In this paper we propose a method to generate semantic maps for robotic indoor localization based on data already available from Building Information Models (BIM). The approach described in the paper aims at defining a method to integrate the rich semantic data-set provided by BIM models to robotics world models, taking as use case indoor semantic localization in a large university buildings. To this end, we approach localization from a model-based perspective. We convert a subset of semantic entities with associated geometry present in BIM models and represented in the Industry Foundation Classes (IFC) data format to a robot-specific world model representation. Such representation is then stored in a spatial database from which the robot can query semantic objects in its immediate surroundings. The robot’s feature detectors are configured and used to detect queried features enabling explicit data associations with semantic objects that are located near the robot’s current position. A graph-based approach is then used to localize the robot, making the map-feature associations explicit and allowing the robot to explain which semantic building features it has seen and used for inferring its location. We show that this model-based approach allows a robot equipped with a 2D LiDAR and odometry to track its pose in a large indoor environment for which a BIM model is available.

I. INTRODUCTION

Mobile robots are deployed more and more in dynamic environments shared with and familiar to humans such as hospitals ([11]), restaurants ([2]) or nursing homes ([3]). Due to the familiarity of the environment, there is an expectation that mobile robots will not only robustly perform long-term autonomous tasks, but that they will be able to understand the environment they operate in in terms of the same semantics that (unskilled) operators, bystanders and engineers use to designate objects in the world. To this end, adding semantics to existing geometric representations used for localization has been an extensively researched topic within the robotics community. The aim of the semantic map that we consider in this work is to provide the link between the semantic perception of the robot and the algorithms that localize it geometrically within these maps. We present a case study for leveraging an existing standard for semantic building representation from the built environment domain for indoor localization. In this study, we use a robot equipped with a conventional 2D laser range finder, which is still to be found on many existing platforms. We show how semantic maps can be used for explainable 2D LiDAR based localization in environments where other sensor modalities may be infeasible due to cost or privacy-by-design is a requirement.

A. Related work

Semantic indoor maps for robotics

Among the various representations for indoor semantic maps used in practice (see, e.g., [4] for a review), we focus on object-oriented representations, as opposed to semantically labeled low-level sensor features, such as pointclouds, or spatial discretizations, such as grid/voxel maps. An overview of indoor map standards used for positioning is given in [5]. Of special interest are the standardized indoor map formats that have been used in robot localization contexts. OpenStreetMap (OSM) is an opensource crowd-sourced mapping initiative that has been extended for indoor robot navigation in [6]. The authors propose an indoor tagging schema to represent the indoor domain hierarchically using the OSM primitives. Objects such as walls and doors are tagged, and related to rooms and hallways. Furthermore, specific areas are represented and tagged to define, e.g., traffic lanes for navigation. The authors show an example of a hospital environment that was modeled and tagged manually using architectural drawings as a reference. The authors acknowledge that the manual creation of the maps using the available Geographic Information Systems (GIS) tools is tedious and that the choice of georeferenced (latitude/longitude) coordinates is not an obvious representation for indoor geometry. An alternative to OSM is provided by indoorGML, which specifically targets indoor spaces and represents geometry, topology and semantics in a multi-layered space model. While indoorGML provides many relevant modeling concepts for indoor navigation, existing work is limited to automatic extraction of indoorGML models from occupancy grid maps ([7]). A different representation of semantic maps in spatial databases is suggested in [8] in the form of the SEMAP framework. In their work, the authors represent semantically annotated 3D object models in a PostGIS database. The models are geometrically represented in PostGIS’ own 3D extension of the “Simple Feature Access” ([9]) specification with the addition of a spatial ontology that allows to perform spatial queries for

\*This work was not supported by any organization

\textsuperscript{1}H. Kwakernaak is with Faculty of Electrical Engineering, Mathematics and Computer Science, University of Twente, 7500 AE Enschede, The Netherlands h.kwakernaak at papercept.net

\textsuperscript{2}P. Misra is with the Department of Electrical Engineering, Wright State University, Dayton, OH 45435, USA p.misra at ieee.org
the map. The authors show multiple applications, including topological location queries of the map for an agricultural harvesting scenario. The authors do not focus on metric robot localization from on-board sensor data and do not consider the representation of uncertainty in their approach. Another work that has similarities to ours is [10], in which the authors perform robust indoor localization with a laser scanner starting from architectural floor plans. They update the floor plans using pose graph SLAM techniques and robust matching criteria. This makes their approach able to perform long term navigation in the presence of map mismatches and environmental disturbances. Contrary to the approach we present, the authors do not focus on semantics or standardized map formats in their work. Furthermore, they aim to provide an updated map of all geometry to match with using ICP techniques, thereby not focusing on semantic features that are known to be static.

In the above works, there is a strong reliance on drawings, maps, and models that either have a geospatial background (e.g. GML, GIS), or rely on manual input based on architectural floor plans. As these sources are not that widely available, nor fully reliable, this makes the use of a laser-scanner device and precise localisation based on point cloud matching with these provided sources critical. Yet, for many buildings, detailed Building Information Models (BIM) ([11], [12]) are currently available, which include 3D geometry as well as detailed semantic data (materialisation, object properties, etc.), and detailed FMIS databases that collect sensor data and occupancy data which can be used to predict expected occupancy in any of the rooms visited by the robot. Such data is being made available at an increasing rate as ‘digital twins’ of buildings [13].

In this work, we want to take advantage of the increasing availability of building information models and digital twins to propose an alternative approach to semantic map creation and representation for the robotics domain. We do so by creating an explicit link between the rich semantic information contained in Building Information Models (BIM) and robotics semantic maps. This combined data source is envisioned to be the basis for a digital twin that can be consumed by robots. Our approach allows to automatically populate a semantic map representation of the operational environment avoiding the manual effort cited by [6]. In addition, the robot’s semantic map and the BIM model use the same semantics and the same reference coordinates to indicate features in the environment which we consider an important step in reaching shared semantic understanding of indoor environments between humans and robots. We validate the proposed approach by means of real life experiments that show how the semantic map derived by BIM models can be used for indoor semantic localization.

**BIM models**

3D BIM modelling software and BIM modelling processes are increasingly taking over the Architecture, Engineering, and Construction (AEC) industry. In many countries, newly built buildings are modelled in BIM software, and delivered to the client as an as-built model (hand-over phase). The operational phase of the building relies less on BIM models, and instead predominantly relies on Facility Management Information Systems (FMIS), which typically have considerably less detailed 3D geometric data, and instead focus heavily on the collection of sensor data (access, temperature, indoor air quality, ventilation devices, etc.). This collection of data is grouped at an increasing rate under the term ‘digital twin’: the digital counterpart of the physical building.

A dominant data standard in the AEC industry is the Industry Foundation Classes (IFC) 1. This standards focuses heavily on the interoperable exchange of 3D data across BIM authoring tools, and therefore also mainly in the phases before the operational phase of a building. An open and neutral IFC file can be exported from a BIM modelling tool, making semantic and 3D geometric data openly available (human- and machine-readable). Even if this data source is used less often for existing buildings [14], it still provides an invaluable resource of information if it is available.

Recent research and standardisation initiatives have aimed at making the IFC data more easily re-usable in the recent ecosystem of software languages and tools. Whereas IFC has always been available in the EXPRESS information modelling language, recent works have aimed elaborately at enabling XML, JSON and RDF formats for the same data. An XML format has been supported since the early 2010s, the RDF format for IFC data is available since 2016 [15], and a simplified JSON format is under construction at the time of research and writing 2.

Previous research has tried to leverage the rich set of (semantic) information provided by BIM models to tackle typical challenges of mobile robots indoor navigation systems such as path planning and localization. With respect to localization, BIM models have been mostly used for indoor image-based localization. Acharya et al. [16] generates a data-set of synthetic images with associated known 6-DOF camera locations and orientations from BIM models. The synthetic data-set is used to train a Deep Convolutional Neural Network (DCNN) which is used for indoor localization. Similarly, [17] generates a data set of synthetic images from the BIM model and trains a DCNN to extract features from

---

1https://technical.buildingsmart.org/standards/ifc/
2https://github.com/IFCJSON-Team/IFC.JSON-4
the generated synthetic data set. Features extracted from the synthetic data are compared with features extracted from images of the physical environment and used to select the synthetic image (with known 6-DOF pose) that is closer to the physical image. Both works differ from what we present in this paper because the localization method is based on camera and a DCNN that has to be trained on the specific building model. Additionally, these papers do not focus on describing a methodology to automatically extract such information from the BIM models. Our approach differs because (a) the BIM model directly provides the features to look at (query of a semantically rich digital twin) without the need to training the DCNN and (b) we present a methodology to automatically extract relevant information from the model avoiding much of the error-prone manual effort. Other work has focused on using the BIM model to derive the topology of an indoor environment from which a path can be planned. In this case, localization is not further elaborated, as opposed to what we aim at in this article. In [18], the authors propose to extract information from BIM models to set-up a simulation environment (VEROSIM) for robotics development. The environment can connect the OMPL (Open Motion Planning Library) to the imported BIM model to generate collision free paths. On a similar line [19] derives a topological graph from BIM models upon which an A* planner can retrieve the optimal path. Although they are valuable and important reference works, our work aims precisely at a live localisation based on a current digital twin of the building model and matching of geometric features.

B. Contribution and contents

In our work, we identify two contributions:

- Proposing a workflow to leverage the semantic and geometric information in building models for indoor robot localization, via automatic composition with a robot-specific property-graph representation in a spatial database.
- Demonstrate the feasibility of the approach for a semantic location tracking task based on semantic features that are known to be static. Explicit data associations are maintained for semantic explainability. We show in an indoor experiment that the semantic approach can be used for a simple 2D LiDAR equipped robot.

The paper starts by introducing the property graph approach for world representation, followed by the step of populating this world model with BIM entities. Second, the localization approach is discussed, which uses existing graph optimization techniques to obtain the robot pose from semantic data associations. An indoor experiment is shown in which a robot localizes indoor based on a BIM model, while giving semantic explanation. Finally, the technical challenges that arise when using existing building models for localization are discussed, and topics for future work are identified.

II. CONNECTING BIM DATA REPRESENTATION TO THE ROBOT WORLD MODEL REPRESENTATION

A. World model representation

We use the term *world model* to describe the robot’s internal representation of itself and its surrounding (i.e., the map), together with knowledge on how to use this information given its capabilities. To accommodate this information, we use a property graph data model which enables composition of different domain models while still keeping a clean separation. This general data model has been used extensively in knowledge representation for robotics (see [20] for an example application). In this work, we focus only on the semantic map and its relation to available sensors. To represent the property graph we use the JSON-LD host language, which provides the mechanisms for attaching a unique symbolic id (@id), model id (@type) and namespace reference (@context) to every entity in the world model. For the geometric primitives in this work, we use 2D “Simple Feature Access” ([9]) representations. We augment these primitives by giving each points that belong to the compositions (such as polygons) all individual ids, allowing to maintain topological consistency. These geometric representations are then associated with the semantic entities by represented_by relations, allowing multiple representations (e.g., 3D geometry, point clouds or images) to be attached in future work. In our approach, the representational shape of an object is an important semantic property of that object that allows to detect it regardless of its position. Furthermore, we explicitly relate representations of objects to the sensors through which they can be perceived. We introduce the ObjectFeatureRepresentation relation that connects the geometry representation, the object and the sensor, allowing the robot to query for features that it can perceive using its sensors.

B. Data conversion

![Fig. 3. Conversion from the BIM model to the representation used by the robot, stored in a database, as used in this work.](image-url)

The property graph model for localization has to be generated from the BIM model for all relevant objects. This procedure is schematically depicted in Figure 3 and will be explained in this section. First, the BIM model is queried for relevant objects, which can appear at different
Fig. 2. Graph representation of the semantic entities from the BIM model and the geometry representation that are perceivable by the sensor, in this case the 2D LiDAR. Different domains (i.e., simple feature geometry, IFC entities and navigational relations) are shown in different colors.

positions in the model hierarchy (e.g., an IfcWall can be declared as part of a space, or as a connected attribute of another IfcWall). For this reason, the BIM model of a floor of the building is exported to an IFC-JSON representation [21] which is then made into valid JSON-LD by adding a @context specifier for the types and relations in the document. This document can then be “framed” by the JSON-LD API [22], turning it into a tree structure where objects of interest are at the root for processing without requiring a graph database and query language. In this paper, we use the IfcColumn and IfcWall entities from the IFC model for localization. These entities are queried from the model together with their representation and objectPlacement relations. For the columns, the sweptArea 2D representation is converted into a polygon profile, for which the local coordinates are converted into global coordinates using the column’s objectPlacement.

For the walls, an IfcPolyline represents the centerline, together with an offset for the thickness. This representation is converted into a slightly different representation with two (inner and outer) polylines, that connect to adjacent wall segments using corner points (see Figure 4). This preserves the correct representation and topology of wall segments and corners.

Fig. 4. Conversion of the BIM line-with-offset representation of a wall (left) to a representation consisting of two polylines for each wallsegment with a shared Point on the corner connecting two wallsegments (right).

The final result is exported as a JSON-LD property graph, which is partly visualized in Figure 2. A small section of the resulting map of a floor of a building is represented in Figure 5. The features are annotated with the type of the semantic object that the features represent.

Fig. 5. Partial view of the map of a floor of the building, as generated from the BIM model, with static features relevant for the lidar sensor. The features are annotated with the types of the objects that they represent.

C. Spatial database and queries

For querying the spatial features and their semantic relations, we use the well-known PostgreSQL database with the PostGIS spatial extension. This database supports both 2D and basic 3D geometry in a Cartesian coordinate system, and offers a rich set of spatial functions and spatial indexes for efficient querying of geometry. We store the property graph representation in the database as well, making it possible to query for relations and entities using the SQL query language. The database is queried for spatial features that are close to the robot. To decide which features are useful, the sensor type is part of the query, resulting in features that are part of an ObjectFeatureRepresentation perceivable by the given sensor. The query returns the feature id, feature type, object id and object type for each feature, together with the spatial object that contains the actual coordinate data structure. This explicit symbolic link between the geometry, its interpretation and the object will be maintained in the association-based localization approach.

III. LOCALIZATION

In this section, we explain our approach to localization. While semantic annotations are not necessary for robot pose tracking (i.e., the geometry suffices), they provide added value for both explaining the robot’s assumptions and understanding the local observations without being reliant on
Fig. 6. (a) Picture of the indoor environment with the robot (b) The same location showing the localization output. Matched semantic features are indicated by grey lines originating from their corresponding robot pose. LiDAR points are in green. (c) An example of the mismatch between the BIM model and the actual environment, making localization more difficult. The location of the square space showed a significant mismatch with reality, causing the robot location to jump. Furthermore, due to glass and doors, not all walls are always perceivable.

Fig. 7. Visualization of the localization approach that queries features from the database, matches sensor features to them and optimized the resulting factor graph.

we have an accurate location estimate all the time. In this work, we only focus on tracking the location and explaining which semantic objects the robot uses to infer its location. While the robot is tracking, it queries semantic features from the database that are perceivable by its on board sensors and are currently within a certain radius around the robot’s location on the map. It then tries to match them with features found in its sensor data. This map-query-first approach allows to more efficiently process sensor data. The current implementation supports the extraction of line features, corner features and box features from the sensor data, based on a split-and-merge weighted line fitting implementation from [23], [24]. The corner and box features are obtained by checking if the lines support the well-known L-shape used often for rectangular objects. In the configuration of the detectors, we demand that the line segments used for wall and corner detection are sufficiently long to be insensitive to open doors. While this threshold is currently manually set to 1.2m, it could be derived from the IfcDoor representations in the BIM model if these are present.

The measurements are added to a factor graph containing the robot poses over a variable horizon, as well as range-bearing measurements to perceived object. For columns and corners we use the well-known range-bearing measurement model. For walls, we use a similar models with the difference that it constrains the angle and distance to the wall, but not the position alongside it. These models have been added to the Appendix. This model allows to use the knowledge that a wall typically does not provide this information, and avoids spurious matches that methods like ICP may suggest. If based on the features suggested by the map, a match in the sensor data is found (e.g., a line segments that falls within the line segment of the wall representation up to a threshold), the feature and measurement get added to the factor graph. We explicitly reference the id of the features in the factor graph, making the data associations with the map explicit. This feature-based approach has benefits over scan matching based approaches such as ICP, because it first checks if the sensor data supports the primitive feature suggested to be use-able by the map. This also allows to disable individual features without removing them from the map by removing their perceivable_by relation. One of the benefits of using graph optimization is that no inconsistencies arise due to linearization. Since the semantic features we use can be relatively sparse, substantial drift corrections can be made which is an important benefit. We do not focus on the tradeoffs of graph-based mechanisms for pure localization in this work, as there are other papers that do this well. See, e.g. [25], [26] for a comparison of factor graphs and particle filters for pure localization. Our main focus is to show that this method makes observations explicit. Exploiting these explicit links for robust navigation tasks or updating the BIM model real-time using SLAM is left to future work. To keep the graph solver efficient we use a moving-horizon approach in which the horizon is adjusted based on the amount of features that have been found in the sensor data and associated with the map. When more than N map features are present in the horizon, we remove the oldest features and their corresponding measurements.
until again $N$ features are present. We have implemented the localization in C++ using the GTSAM [REF] optimization library together with our own in-memory semantic graph model and bookkeeping functions.

IV. EXPERIMENTS WITH A 2D LIDAR-EQUIPPED ROBOT

In this section, we show an experiment with a mobile robot in a building for which an IFC model is available (visualized in Figure 1). We use a custom-made platform equipped with mecanum wheels, wheel encoder odometry and a Hokuyo UTM30-LX 2D LiDAR scanner mounted upside down, close to the floor. Due to its mounting position, we have a $180^\circ$ field of view consisting of 720 scan points. We run a total of three trajectories (approx. 100 meters each) with the robot by tele-operation, starting from different initial positions. These positions are provided to the localization algorithm by initial pose estimates. The environment is cluttered with both semi-static objects (furniture, plants) and dynamic objects (chairs, carts), as shown in Figure 6a. In the third dataset, three actors are walking around in the field of view of the robot. The robot’s feature detection is triggered whenever a set distance has been driven and the map is first queried for features that are visible to the 2D LiDAR within a range of 6 meters. The sensor data is then processed to search for sensor features that support the object features on the map and the pose estimate is update according to the approach explained in the previous section.

Figure 9 shows the 2D map with the features perceivable by the 2D LiDAR and the three trajectory estimates resulting from our localization approach. The horizon length is truncated at all times based on three semantic object references. Figure 6b shows an example of the horizon in which these references are visualized by edges (grey lines). Using our approach, the robot was able to successfully track its location by incrementally associating with objects on the map generated from the BIM model. In the next section, the results are discussed in more detail.

V. RESULTS AND DISCUSSION

The object types that are matched with the building model are visualized in Figure 10. Both the number of associated measurements with a object of that type and newly detected objects entering the horizon are depicted. The resulting trajectories are shown in Figure 9. The features used for localization in our approach were selected because of their availability in the BIM model and their saliency. No false positive associations were made in our experiments. We selected these features because we predict that this saliency carries over to other indoor environments as well. In figure 6 it can be seen that especially the wall features are very strong features that, once visible, have a high recall and are spotted again within the horizon. The result is a graph which contains a lot of variables. The column features are consistently spotted as well, but because of the L-shaped visibility requirement, require a more specific viewpoint. Again this approach was deliberately chosen to favor precision over recall performance. The corner features are less strong features, because of the same L-shape detection requirement. Spurious corner measurements can easily give rise to inconsistent matches and is avoided by favoring precision. Furthermore, from Figure 6 we can see that columns and wall are both necessary for consistently having a set of new features entering the horizon. This brings up to the question whether our method is easily transferable to different environments. This of course depends on the availability and contents of the BIM model available for another environment. However, we emphasize that also the semantic explainability of local detections is an important requirement in our work, that will come to its full right when exploited in context of different robot tasks, such as navigational rules (e.g. driving close to walls or using a column as natural waypoint).

Another observation from our experiments is that while building model have extensive support for representing object properties and geometry, the models are not always accurate or complete. The environment we tested in contained glass walls that were marked as regular wall segments in the BIM model. Furthermore, significant spatial inaccuracies were present in the, one of which is shown in Figure 6c. These inconsistencies may require specific strategies to deal with them that we did not employ in our work, such as the use of discrete hypotheses, or updating the map. However, robustness to dynamic environments is also one of the advantages of our approach, because only features that are known to be static are used for localization. An updated map that contains non-semantic additions will contain dynamic features and therefore might introduce new failure modes. A final remark is that the relative inaccuracy of the BIM model raises an import question about the definition of accuracy. Do we define accuracy as the metric deviation of the position in a map coordinate system, or do we define it as the correctness of perceiving local objects of interest with respect to the robot. The latter enables a definition of accuracy in the case were supplied maps are not perfect but localization can be robust nonetheless.

VI. CONCLUSIONS AND FUTURE WORK

In this work, we showed that existing semantic building information models provide a robot with enough information to localize itself, providing great opportunity for automatic
deploymnt in large buildings without prior work or adaptation. We also showed how this semantic information can be translated to explainable detections and implemented for a 2D LiDAR equipped robot, using a property graph database to let the robot query its semantic input. The opportunities we see for future work will consist of generalizing our approach to different semantic entities that are present in BIM models and using different sensors to perceive them. Furthermore, keeping the BIM map up-to-date and consistent is an important challenge. Our semantic approach explicitly enables to select features that are reliably detectable form the BIM model and only mark those as perceivable and thereby useable for localization. The next step would be to update geometry of the BIM model, which requires clear policies of data governance. Moreover, we consider the data provided through the BIM model as the start or basis of a digital twin (database) that can also be used in the operational phase of the building, and which is enriched with plenty of operational data that can be of use for robot navigation (e.g. crowds). To conclude, we foresee that automatic deployment of robots in buildings is can be useful in many scenarios and our work has explored important steps in making this a possibility.

VII. APPENDIX

For columns and corners we use the well-known range-bearing measurement model:

\[
\begin{align*}
    Z_\rho &= \sqrt{(x_\ell - x_r)^2 + (y_\ell - y_r)^2} + \epsilon_\rho \\
    Z_\theta &= \arctan2(y_\ell - y_r, x_\ell - x_r) - \theta_r + \epsilon_\theta
\end{align*}
\] (1) (2)

For walls, we use the model:

\[
\begin{align*}
    Z_\rho &= (\cos(\theta_r) * (p_1x - x_r) \\
        & \quad - \sin(\theta_r) * (p_1y - x_r)) * \cos(Z_\theta) \\
        & \quad + (\sin(\theta_r) * (p_1x - x_r) \\
        & \quad + \cos(\theta_r) * (p_1y - y_r)) * \sin(Z_\theta) + \epsilon_\rho \\
    Z_\theta &= \arctan2(A, B) + \epsilon_\theta
\end{align*}
\] (3) (4)
where $A$ and $B$ are given by
\begin{equation}
A = \cos(-\theta_r) \ast (p_{2x} - p_{1x})
- \sin(-\theta_r) \ast (p_{2y} - p_{1y})
\end{equation}
\begin{equation}
B = \sin(-\theta_r) \ast (p_{1x} - p_{2x})
+ \cos(-\theta_r) \ast (p_{1y} - p_{2y})
\end{equation}

This model constraints the angle and distance to the wall, but not the position alongside it. In this equations $(x_r, y_r, \theta_r)$ denotes the pose of the robot in the map. A measured point is represented by $(x_t, y_t)$. The measured line is represented by its two points $(p_{1x}, p_{1y}, p_{2x}, p_{2y})$. The measurement noise terms $\epsilon_x$ and $\epsilon_y$ are Gaussian additive terms. Note that in the graph-based approach it is possible to separate measurement noise and map uncertainty, by making the locations of the measured points uncertain variables with corresponding prior densities. This allows to solve for - and update - the MAP location of these points.

REFERENCES

[21] IFCISON-Team/IFC.JSON-4: Repository containing the specification for IFC.JSON.
[22] JSON-LD 1.1 Framing.
[23] GitHub - kam3k/laser_line_extraction: A ROS package that extracts line segments from LaserScan messages.