Semantic navigation with domain knowledge

Rafael Gomes Braga¹ · Sina Karimi² · Ulrich Dah-Achinanon³ · Ivanka Iordanova² · David St-Onge¹

Abstract Several deployment locations of mobile robotic systems are human made (i.e. urban firefighter, building inspection, property security) and the manager may have access to domain-specific knowledge about the place, which can provide semantic contextual information allowing better reasoning and decision making. In this paper we propose a system that allows a mobile robot to operate in a location-aware and operator-friendly way, by leveraging semantic information from the deployment location and integrating it to the robot’s localization and navigation systems. We integrate Building Information Models (BIM) into the Robotic Operating System (ROS), to generate topological and metric maps fed to an layered path planner (global and local). A map merging algorithm integrates newly discovered obstacles into the metric map, while a UWB-based localization system detects equipment to be registered back into the semantic database. The results are validated in simulation and real-life deployments in buildings and construction sites.

Keywords Semantic navigation · BIM/IFC · Map Merging · Path Planning

1 Introduction

As mobile robots are deployed in the real world to perform high level tasks in complex scenarios, possibly around humans, it becomes increasingly interesting to integrate rich data from the environment into the robot’s systems. While current navigation techniques based on geometric information allow the robot to navigate safely and efficiently in challenging environments, applications such as self-driving vehicles, service robots and automated construction progress monitoring require a higher level of reasoning and decision making by the robot. For example, a robot’s task could be “navigate from the hall to the kitchen”, in which case a framework that integrates the room identity along with the geometric information is required.

The relatively new field of semantic mapping tackles this problem by investigating ways to generate maps that contain the geometric and semantic information of the environment, usually combining semantic segmentation techniques with SLAM algorithms. Semantic information is extracted from sensors such as depth cameras in Kochanov et al. (2016) and LiDAR in Chen et al. (2019). Most current research employs deep learning methods, for example McCormac et al. (2017) and Xi-ang and Fox (2017), with very promising results. These approaches however are complex, computationally demanding and require a large amount of data in order to train the classification models.

On the other hand, most buildings nowadays possess some form of digital documentation, with Building Information Model, or BIM, being the most promi-
The current paper contributions are as follows:

- An optimal high-level path planner integrated with the low-level navigation (cascade navigation stack);
- A flexible semantic teleoperation and navigation for an autonomous UGV in human made structures;
- A practical implementation of the whole system, including active equipment detection, deployed on an autonomous mobile robot navigating a construction site.

The work is organized as follow. Section 2 summarizes the inspirational works to our approach. Section 3 describes the generation of topological maps (hypergraphs) from BIM information. Section 4 details the path planning algorithm. Section 5 explains how onboard maps are merged to the semantic map. Section 6 describes our equipment detection approach based on Ultra-Wide Band (UWB) localization. The simulations and field deployment used to validate the proposed system are described in section 8. The results of our experimental validation are discussed in section 9. Finally, section 10 summarize the contributions and the next steps of our work.

2 Related work

Conventional methods of indoor path planning often refer to the optimal path as the shortest path calculated by various algorithms such as A* and Dijkstra’s (Palacz et al. 2019). However, short and safe (avoiding collisions) attributes are often not enough to address complex navigation tasks. In human-made infrastructure, Building Information Modeling can provide the robot with elements’ semantics and geometries (Karan and Irizarry 2015) increasing its understanding of the mission. Many studies suggested ways to leverage BIM for indoor navigation. Wang et al. (2020) develop a framework to convert the BIM digital environment to a cell-based infrastructure to support indoor path planning. In their work, they emphasize on the “BIM voxelization” process leaving the path planning problem undressed. In Song and Marks (2019), a BIM-based path planning strategy is used for equipment transport on construction sites. The authors extract the start and end points from BIM and then generate the shortest sequence of rooms for the operator, but do not support local-level adaptive robot path planning. Ibrahim et al. (2017) propose a path planning strategy based on BIM for an Unmanned Aerial Vehicle (UAV) on construction sites but only for outdoor usage. Follini et al. (2020) proposed a UGV-based logistic support system with a human-assisted approach enhanced by a metric map extracted from BIM, but all experiments were performed in a controlled environment and the BIM semantics are not thoroughly leveraged. In Ibrahim and Golparvar-Fard (2019), the optimal route for a data collection mission using an UAV is proposed, leveraging 4D BIM to identify which building spaces are expected to change, but implemented only in a simulated environment.

Dellbrügger et al. (2017) developed a framework supporting humans and autonomous robots navigation which
mostly uses building geometries in a simulated environment. In [Nahangi et al., 2018], the indoor localization of an UAV is assessed using calibrated visual markers (known location in the BIM). The markers setup is a tedious step that will not fit most deployment scenarios. Another study focused on the use of BIM for robot localization with hierarchical reasoning for path planning [Siemiatkowska et al., 2015]. Along that line, BIM was also demonstrated to be powerful for the identification of various paths through a graph strategy [Hamieh et al., 2020]. An approach using hypergraphs generated from IFC files was also developed in which a modified A* algorithm is able to detect the optimal path among the rooms [Palacz et al., 2019]. These works provide only high-level path (rooms sequence) with respect to BIM geometries and lack the integration with a standard robotic architecture.

For active equipment detection, we extend the vast literature on UWB usage for localization. For instance, [Yin et al., 2016] presented an indoor positioning system, based on the UWB technology. That work applied the trilateration localization algorithm to find the position of a blind node based on range measurements and known UWB anchors’ positions. The principles of the algorithm are transferable to our use case where we are doing the exact opposite, i.e., finding the position of the anchors based on range measurements and the position of the blind node. The beacon localization case has also been studied by [Sato et al., 2019] where they localized BLE beacons using RSS observations. For the localization, a person moves around the room to obtain several RSS observations. Like us, they use a range-only and inter-beacon relative distance of twin beacons approach but without a priori information, so they employ a full EKF-SLAM algorithm in order to obtain the person’s location.

In these inspiring works three aspects of the domain knowledge potential for indoor robot path planning are yet to be thoroughly studied: (1) considering the full potential of the building semantic rather than only the geometry (2) integrating the high-level (rooms sequence) with the low-level sensor-based information in a synergistic navigation stack (3) validating in the field the performance of using semantic information. In this paper, we cover these gaps by integrating Building Information Robotic System (BIRS) into a navigation system in ROS for autonomous navigation and intuitive teleoperation.

3 BIM-based topological maps

Topological maps are a conventional method for robot navigation that can be fed with building-related data. In these inspiring works three aspects of the domain knowledge potential for indoor robot path planning are yet to be thoroughly studied: (1) considering the full potential of the building semantic rather than only the geometry (2) integrating the high-level (rooms sequence) with the low-level sensor-based information in a synergistic navigation stack (3) validating in the field the performance of using semantic information. In this paper, we cover these gaps by integrating Building Information Robotic System (BIRS) into a navigation system in ROS for autonomous navigation and intuitive teleoperation.
A path with lower weight is better. To help the robot avoid potential sensor erratic behavior, the material properties of the walls are included in the hypergraph through IfcMaterial and its super-type IfcProduct. The weight of each curtain wall, i.e. invisible walls, in each node is \( w_m = 12 \), and all others are \( w_m = 4 \) since they can be easily detected. To include the passing time from one node to the other, rooms area are also included in the hypergraph, i.e. bigger rooms take more time for the robot to cross. Accordingly, the weight for the rooms less than 50\( m^2 \), between 50\( m^2 \) to 100\( m^2 \) and more than 100\( m^2 \) are \( w_a = 2 \), \( w_a = 8 \) and \( w_a = 12 \) respectively. To increase the efficiency of data collection, the nodes are attributed a scanning age (time since the last scan) to maintain the building knowledge up-to-date. It may be beneficial for the robot to visit more rooms and collect more data. Industry needs define the frequency of data collection, therefore, we assign \( w_s = 10 \), \( w_s = 6 \), \( w_s = 0 \) for the scanning period of less than 1 week, between 1 week and 2 weeks, and more than 2 weeks respectively. To consider robot’s safety while navigating, each node is also attributed with risk level (i.e. construction activity). For instance a high risk would be attributed \( w_r = 500 \). In this case, any other available path should be selected by the algorithm. If there is not an alternative safe path for the robot, the algorithms provides a warning for high-weight paths so that the supervisor of the robotic deployment is warned. We use directed hypergraph (with directed hyperedges) allowing us to assign cost for door opening directions. IfcDoor as a sub-class of IfcBuildingElement provides the center coordinates of the doors creating hyperedges (with their coordinates) in the hypergraph. IfcDoor also stores the opening direction through y-axis of ObjectPlacement parameter. For pushing and pulling the door, we assign \( w_d = 2 \) and \( w_d = 6 \) to the hyperedge’s weight respectively.

### 4 Finding the optimal indoor path

The process of generating the hypergraph is done with a Dynamo Script (a visual programming tool) to extract the IFC classes and their parameters. Then, the information is stored in a XML database to be retrieved by a Python script. In order for the data to be ROS-friendly, the Python script translates the data. Having the hypergraph of the building, start and end nodes (rooms) can be defined by the user and the optimal path is automatically generated by the algorithm. We implemented directed BF-hypergraph since each room (node) can be connected to multiple rooms [Gallo et al., 1993]. Since door opening direction is considered for assigning weights to the hyperedges, each pair of nodes is connected forward and backward. This creates backward and forward sub-hypergraphs within the overall topological map. "Shortest Sum B-Tree" is used to find all the possible hyperedges between the start and end nodes. The hyperedges contain the weight of passing from one node to the other. Having a set of paths from start to end nodes with their weights enables the "Shortest Sum B-Tree" to select the path with the lowest cumulative weight. Since the hypergraph represents the building semantics, all the information for semantic navigation is provided to the user. As illustrated in fig. 2 the optimal path outputs a set room names, their coordinates and a set of door coordinates in the sequence of node location and hyperedge (door) location.

### 5 Probabilistic map merging with semantic data

The previous step can feed a standard onboard navigation stack to plan a path using a LiDAR, the known layout, and an algorithm such as A*. However, that would leave out all the semantic knowledge still available from the building model. In order to properly locate the robot in the building layout while updating that layout from the mission measurements, we developed a probabilistic map merging algorithm that takes into account the semantic data. It is composed of a localization module based on a particle filter that finds the robot’s pose in the a priori map using a laser scan probabilistic model that changes the expected sensor values based on the semantic classes of the objects in the environment. In this section we explain the implementation in detail.

We consider a robot moving in the 2D plane with pose \( x_t \) at time \( t \) given by:

\[
x_t = \begin{pmatrix} x & y & \theta \end{pmatrix}^T
\]  

(3)
Fig. 3: Semantic Graphical User Interface for the intuitive operation of a UGV with domain knowledge. The controls in the header allow selecting a destination and generating the path. The panel to the left shows the attributes of the selected room. The center contains a map of the environment, with the robot’s pose in real time being represented by the purple arrow. The center points of the rooms and doors in the path are represented in the map by the yellow circles. The right panel allows the user to reconfigure the different weights that are applied to the path generation.

The environment is represented by a map $m$ as a grid of cells $m_{i}$, $i \in \{1,...,N\}$ where $N$ is the number of cells. Each cell has an associated tuple $(p_i, c_i)$ where $p_i \in [0,1]$ is the occupancy value of that cell and $c_i \in \{0,1,...,C\}$ is an integer representing the class to which that cell belongs. For occupied cells, the class refers to the material that surface is made of, as identified by a robot equipped with the necessary sensors. If the material can’t be identified, we use the value 0, same for unoccupied cells.

Assuming we start with an a priori map extracted from BIM, we first want to estimate the pose of the robot relative to the map’s reference frame. Similar to the Monte Carlo Localization algorithm described in Thrun (2002), we construct a particle filter where each particle $(i)$ is a tuple $(x_{t}^{(i)}, w_{t}^{(i)})$ such that $x_{t}^{(i)}$ is the robot pose and $w_{t}^{(i)}$ is the particle weight at time t. For each control/measurement pair, the particles are updated according to the following equations:

$$x_{t}^{(i)} = M(u_{t-1}, x_{t-1}^{(i)})$$
$$w_{t}^{(i)} = S(z_{t}, x_{t}^{(i)}, m)$$

Where $M$ is a motion model and $S$ is a measurement model, as defined in Thrun (2002). To mitigate misdirection caused by inconsistencies between the map and the sensor data we modified the sensor model to account for the class information associated with each cell in the map. This information is encoded in the map when the semantic data is extracted from BIM. Below we describe our measurement model:

The measurement model is defined as a conditional probability distribution $p(z_{t}|x_{t}, m)$, where $x_{t}$ is the robot pose, $z_{t}$ is the measurement at time $t$ and $m$ is the map of the environment.

For a 2-D laser range finder with $K$ beams, $z_{t}$ is a collection of measurements $z_{k}^{t}$ where $k \in \{1,...,K\}$.

The probability $p(z_{t}|x_{t}, m)$ is obtained as the product of the individual measurement likelihoods:

$$p(z_{t}|x_{t}, m) = \prod_{k=1}^{K} p(z_{k}^{t}|x_{t}, m)$$

A typical laser range finder model is a mixture of four densities, each corresponding to a different error the sensor measurements are subject to. Figure 4 shows a visualization of the resulting density, where $z_{max}$ is the maximum sensor range and $z_{k}^{t}$ is the true value that would be measured by beam $k$ in a ideal world, given the robot pose $x_{t}$ and the map $m$. This value can be obtained by a ray tracing algorithm, such as Bresenham’s line algorithm [Bresenham, 1965], which draws a line starting on the robot’s pose and moving outward, according to the beam’s angle. Once an occupied cell is found, it returns the distance between the robot and that cell. In our model, we also take into consideration the class information: the algorithm only returns if the occupied cell also belongs to a class that can be detected by the sensor. This allows the estimator to ignore obstacles that wouldn’t be detected by the current sensor, like a glass wall and a laser scanner.
L covered using: the log odds notation. To simplify the calculation and avoid instabilities, we use applying the Bayes filter described in Thrun (2002). To readings can be combined to the occupancy map by algorithm, as shown in Thrun (2002).

Fig. 4: Probability density for a beam of a 2D laser scan sensor. $z_{\text{max}}$ is the maximum sensor range and $z_i^{k+}$ is the theoretical true value obtained from a ray casting algorithm, as shown in Thrun (2002).

Once the robot’s pose is obtained, the current sensor readings can be combined to the occupancy map by applying the Bayes filter described in Thrun (2002). To simplify the calculation and avoid instabilities, we use the log odds notation.

$$L(m_i | z_{1:t}) = L(m_i | z_{1:t-1}) + L(m_i | z_t)$$

(6)

The actual occupancy probabilities can be easily recovered using:

$$L(m_i | z_{1:t}) = \log \frac{p(m_i | z_{1:t})}{1 - p(m_i | z_{1:t})}$$

(7)

6 UWB-based equipment localization

While the robot moves around and update its map, it can detect active equipment that need to be tracked on site. Our solution to this challenge requires ranging measurements performed between the robot and the beacons placed on the equipment. We selected UWB technology for our beacons: a radio technology that uses a very low energy level for short-range communications and that operates at high frequencies (3.1 to 10.6 GHz). That choice was mainly motivated by the accuracy of the UWB technology and its robustness to occlusions. In fact, according to Jiménez and Seco (2017), it has a 10 cm precision while the BLE can go up to 5 meters.

We are thus trying to locate UWB beacons attached to equipment based on known robot positions. To do so, we adapted the concept of trilateration localization algorithm coupled with an optimization algorithm.

More commonly, trilateration serves to localize a robot based on known anchors position as presented in Yin et al. (2016). We exploit the same algorithm in doing the exact opposite for our use case, i.e., localize anchors based on known robots positions.

In our use case, the robot is assumed to be moving on a 2D world, hence, the localization is done in 2.5D (with a known height for the anchors). To have a noise resistant system we used 70 known robot positions spread around the anchor. We then get a system of equation that can be solved in matrix form with a pseudo-inverse minimizing the root mean square error:

$$x = (A^T A)^{-1} A^T b$$

(8)

With

$$A = \begin{bmatrix}
2(x_n - x_1) & 2(y_n - y_1) \\
2(x_n - x_2) & 2(y_n - y_2) \\
... & ...
\end{bmatrix}$$

(9)

and

$$b = \begin{bmatrix}
x_1^2 - x_n^2 - y_1^2 - y_n^2 + x_n^2 - 2x_n(z_n - z_1) \\
x_2^2 - x_n^2 - y_2^2 - y_n^2 + x_n^2 + y_n^2 + z_n^2 - 2x_n(z_n - z_2) \\
... \\
x_{n-1}^2 - x_n^2 - y_{n-1}^2 - y_n^2 + x_n^2 + y_n^2 + z_n^2 - 2x_n(z_n - z_{n-1})
\end{bmatrix}$$

(10)

With the first approximation obtained from the trilateration, we use the Trust Region Reflective algorithm to refine the solution. The algorithm solves a minimization problem based on a given cost function and an initial guess for the solution. The cost function here is the sum of the differences between the range measured by the UWB device and the same distance based on the estimated solution, i.e.:

$$\text{error} = \sum_{i=1}^{n} |r_i - (||x - \text{robot}_i||)|$$

(11)

with $r_i$: the range measured by the UWB device at $t = i$, $x$: the solution tested by the minimization algorithm, $\text{robot}_i$: the robot position at $t = i$. 

Rafael Gomes Braga et al.
7 Semantic graphical user interface

A Graphical User Interface (GUI) was developed based on BIM semantics to allow users to intuitively operate the robot and configure the path planner. The GUI connects to the ROS running in the robot and presents semantic information of the building and data from the robot in real time. The integrated high-level and low-level navigation system moves the robot to the desired destination. The GUI allows the non-expert users to work with their domain knowledge, thereby making robot deployment more intuitive and simpler. Figure 3 illustrates the interface window. The GUI is developed in Python notebooks, allowing for easy integration of visualization widgets and customization.

The GUI provides the building’s rooms in a drop-down list, from which the user selects a destination and then launch the path planner to find the optimal path. The center area of the GUI shows a map of the building, with the robot’s pose being updated in real time, along with the paths objectives. The left panel shows the selected room’s (end node) attributes. The right panel allows the user to alter the weights of each parameters of the path planner. After changing and saving the new weights, the user can generate the path again and see the results on the map. Finally, the user can click on the Move Robot button to trigger the robot to start moving.

8 Experiments

We validated our approach in simulation and with an experimental case study, where the system was implemented on a real mobile robot deployed to the field. The experiments were designed to allow the robot to navigate through the corridors of a building while recording data from its sensors. We choose one of the buildings at École de Technologie Supérieure, for which a complete BIM model was available. An operator commands the robot by choosing destinations in a semantic graphical user interface running on a computer, which then sends this destination to the robot through a wifi link. The hierarchical path planner running on the robot generates a list of waypoints to the destination and an internal controller moves the robot accordingly. In the next sessions we describe each of those parts in detail.

8.1 Robotic platform

The robotic platform used in this study is built from a four-wheeled unmanned ground vehicle (Clearpath Jackal) equipped with a hybrid vision/laser scan sensing system and is shown in fig. 5. This robotic system is capable of collecting data while the robot is moving and can navigate semi-autonomously, with a remote operator sending high level commands.

The Jackal is equipped internally with wheel encoders and an IMU, which are used for odometry estimation and control by an embedded computer. Additionally, a NVidia Jetson AGX Xavier computer was installed to process the data and the navigation algorithms. A suite of ROS nodes for control, state estimation and diagnostics are provided by Clearpath, which serve as a basis for the robot operation.

The sensing system was envisioned for data collection in construction sites and contains LiDARs and cameras positioned in different directions to cover as much of the robot’s surroundings as possible. A front-facing Intel Realsense T265 tracking camera combines the mobile platform’s wheel odometry with visual data and outputs high accuracy odometry estimation to the localization system. Another front-facing camera, a Intel Realsense D435i depth camera feeds depth images to a collision avoidance node that reacts when the robot moves too close towards an obstacle. A Velodyne Puck 32MR Lidar is mounted horizontally to scan the walls and floor and is used to detect obstacles and objects by the localization and map merging algorithms. The other sensors are integrated for data collection and used in other studies.

Figure 6 gives an overview of the system. The robot pose in the map is obtained through the use of a ROS implementation of the Adaptive Monte Carlo localization algorithm from [14]. Before deploying the robot, wall geometry information is extracted from BIM to generate an occupancy grid of the building. During the robot navigation, this map, the odometry, and the laser scan data from the horizontally mounted Velodyne LiDAR are fed to the localization algorithm, which then estimates the robot’s current pose in that map. When
Fig. 6: System Overview: A high level planner that processes BIM/IFC information and user inputs is integrated to a low level navigation stack in a cascade design. The low-level module takes care of the localization, local path planning and collision avoidance tasks, while the high-level planner generates paths based on BIM/IFC semantics.

8.2 Simulation

The simulation was performed using the Gazebo Simulator. The building information is exported to create a 3D model, a digital twin. Clearpath, Gazebo and the ROS community provide all the required software packages required to generate an accurate simulation of our robotic platform. Figure 7 shows the simulated robot and its environment with different wall textures and transparency.

9 Results

9.1 UWB-Beacons Localization Validation

To evaluate the active equipment localization algorithm, we performed a series of tests with the Jackal robot, which was equipped with a Pozyx tag placed 78 cm above the floor level. Four UWB beacons representing equipment were positioned in the laboratory area, according to table 1. Figure 8a shows a simplified plan of the test room where the boxes A and B are 1.65 m high desks and the boxes 1 to 3 are 70 cm high.

As discussed previously, the positions considered for the 2D trilateration have to be non-colinear, otherwise the algorithm will output erroneous results. For that reason, the robot’s trajectory must be winding in order to ensure it. The algorithm also enforces a 10 cm interval between consecutive positions used in the algorithm. Figure 8b shows an example of a winding trajectory on a generated plan of the test room.

After a set of 20 experiments (20 detections per anchor) we obtained the results presented in table 2. That results show that the mean error on the detection is around 11 cm on both x and y axes. The worst results comes from the tag 36 which is partially occluded by a desk. The minimal error is about 1 cm while the maximal goes up to 63 cm. Figure 8a also presents a graphical representation of the error and standard deviation for each tags. The green dots represent the ground truth, the black dots are the mean position detected by the algorithm and the blue ellipses represent the standard deviation of the detections.
9.2 Navigation

The experiment had two main objectives:

1. Test the effectiveness of the semantic path planner in generating the optimal path to reach the destination, given the building information obtained from BIM/IFC.

2. Test how changes in the building information affect the final path that is generated.

In our case study, the robot starts in a corridor (CORRIDOR OUEST) on the west side of the building and must reach an open area (CORRIDIR EST) on the eastern part of the building. Figure 9a shows the building map, and the path in red line generated by applying the A* algorithm from start to end. This is the shortest possible path between the two points, taking into consideration only the building geometry and a small safety collision radius around the robot. When the Semantic Path Planner is applied to the same scenario, a similar result is obtained as expected, represented by the yellow path in fig. 9a. Since there are no doors, undesirable materials or hazards in the path, the algorithm outputs a list of rooms that must be visited by the robot that represent the shortest distance from start to end. The semantic path planner provided the order of rooms’ names from the start to the end as it is show in the GUI in fig. 9a Therefore, the user operating the robot can intuitively track the path from the data collected. In this direction, the as-built data can be directly compared to the as-planned since the path is recorded semantically. Also, the waypoints of rooms’ center coordinates and doors’ center coordinates are provided by the semantic path planner. If there is a door made of materials invisible to sensors (such as glass), the complementary door coordinates helps for safer, smarter, precise data collection. Following this, the A* algorithm finds the shortest path between the waypoints.

In a second run, the building information was altered to include a construction operation carried out in the area highlighted with a dashed box in fig. 9b (not visible in the GUI). Since the construction activity represents a hazard with a high cost for the Semantic Path Planner, a different path passing through another corridor is automatically selected, as illustrated by the orange path. Nevertheless, the high cost of the shortest path triggered a warning in the system indicating a hazard to the user through the semantic GUI. Therefore, the user can understand the risks associated with navigation through an active construction area and decide whether to scan the environment or postpone it to a safer time. The yellow path was automatically generated, although it is not the shortest path, as the optimal path from the default parameters mentioned in section 9. This path passes along a large curtain wall invisible to the robot’s sensors. The additional semantic information provided by the BIRS is given to the robot as well as the BIM occupancy grid so it contributes to collision avoidance with the wall. The GUI provides the user with the scan aging of the rooms so the user can decide which rooms to select as the destination for data collection. This allows the users to run multiple data collection mission with the robot which increases the efficiency of robot deployment on construction sites.

For a final run, the robot was commanded to navigate to the destination and then return to the origin. As shown in fig. 10a, the red path generated by A* passes twice over the same corridor, minimizing the ex-
Fig. 9: High-level and low-level paths: A* generates the shortest path possible between start and end, not taking advantage of the BIM/IFC semantics. (a) Without any special condition, both algorithms generate the shortest path. (b) When a construction activity is happening, our semantic path planner is able to find an alternative path, while A* causes the robot to navigate through the hazardous area.

Fig. 10: BIM/IFC semantics can be used to increase explored area. (a) By searching for the shortest possible path, A* causes the robot to navigate the same corridor twice. (b) The semantic path planner avoids recently explored areas, resulting in paths that cover more of the environment.

Our semantic path planner, however, employs BIM/IFC semantics to track the amount of time since a room was last visited and tries to avoid recently visited ones. Figure 10b shows that the path generated when the robot is returning is different from the first one, encouraging the robot to seek unexplored zones. The result is a bigger portion of the environment is covered and more data is obtained by the robot. As illustrated, the integrated BIM-ROS information provides a cascade navigation system on construction sites enabling autonomous and accurate data collection of the spaces scanned.

10 Conclusion

This paper presented a semantic path planner that uses building information from IFC data schema to generate optimal paths for safe and efficient navigation of autonomous robots on job sites during the construction phase. We used the BIRS for extracting building information from IFC represented in a hypergraph structure. Path planning algorithms can then be used to calculate optimal paths in this graph given the building information. Weights are designated to each connection in the path to represent how different conditions can affect the

robot’s navigation and to prioritize paths with more desired characteristics. The optimal semantic path is then integrated with low-level navigation system and A* algorithm is used to calculate the shortest path within the optimal path. The effectiveness of the path planning to generate different paths given different conditions was shown in a simulated and real life case study.

This algorithm can be extended in the future to take into consideration Mechanical, Electrical and Plumbing (MEP) semantics for data collection. Different locations can be added based on the kind of information needed at a specific time of construction through the GUI in order to provide the robot more destinations to collect data. Therefore, the high-level path planning algorithm would provide a more efficient route for data collection as well as semantic navigation. Also, this paper provided semantic navigation of mobile robot on construction sites, therefore, a user study will be conducted in order to assess the usability of the semantic navigation approach.

Acknowledgements  The authors are grateful to Mitacs for the support of this research as well as Pomerleau; the industrial partner of the ÉTS Industrial Chair on the Integration of Digital Technology in Construction.

References

Bresenham JE (1965) Algorithm for computer control of a digital plotter. IBM Systems journal 4(1):25–30
Chen X, Milioni A, Palazzolo E, Giguere P, Behley J, Stachniss C (2019) Suma++: Efficient lidar-based semantic slam. In: 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), IEEE, pp 4530–4537
Delbrügger T, Lenz LT, Losch D, Roßmann J (2017) A navigation framework for digital twins of factories based on building information modeling. In: 2017 22nd IEEE International Conference on Emerging Technologies and Factory Automation (ETFA), IEEE, pp 1–4
Follini C, Terzer M, Marcher C, Giusti A, Matt DT (2020) Combining the robot operating system with building information modeling for robotic applications in construction logistics. In: International Conference on Robotics in Alpe-Adria Danube Region, Springer, pp 245–253
Gallo G, Longo G, Pallottino S, Nguyen S (1993) Directed hypergraphs and applications. Discrete applied mathematics 42(2-3):177–201
Hamieh A, Makhlouf AB, Louhichi B, Deneux D (2020) A bim-based method to plan indoor paths. Automation in Construction 113:103120
Hart PE, Nilsson NJ, Raphael B (1968) A formal basis for the heuristic determination of minimum cost paths. IEEE transactions on Systems Science and Cybernetics 4(2):100–107
Ibrahim A, Golparvar-Fard M (2019) 4d bim based optimal fligh planning for construction monitoring applications using camera-equipped uavs. In: Computing in Civil Engineering 2019: Data, Sensing, and Analytics, American Society of Civil Engineers Reston, VA, pp 217–224
Ibrahim A, Roberts D, Golparvar-Fard M, Bretl T (2017) An interactive model-driven path planning and data capture system for camera-equipped aerial robots on construction sites. In: Computing in Civil Engineering 2017, pp 117–124
Jiménez AR, Seco F (2017) Finding objects using uwb or ble localization technology: A museum-like use case. In: 2017 International Conference on Indoor Positioning and Indoor Navigation (IPIN), IEEE, pp 1–8
Karan EP, Irizarry J (2015) Extending bim interoperability to preconstruction operations using geospatial analyses and semantic web services. Automation in Construction 53:1–12
Karimi S, Iordanova I (2021) Integration of bim and gis for construction automation, a systematic literature review (slr) combining bibliometric and qualitative analysis. Archives of Computational Methods in Engineering pp 1–22
Karimi S, Iordanova I, St-Onge D (2021) An ontology-based approach to data exchanges for robot navigation on construction sites. arXiv:2104.10239 pp 1–21, 2104.10239
Kochanow D, Ösep A, Stickler J, Leibl B (2016) Scene flow propagation for semantic mapping and object discovery in dynamic street scenes. In: 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), IEEE, pp 1785–1792
Kostavelis I, Gasteratos A (2017) Semantic maps from multiple visual cues. Expert Systems with Applications 68:45–57
McCormac J, Handa A, Davison A, Leutenegger S (2017) Semantic fusion: Dense 3d semantic mapping with convolutional neural networks. In: 2017 IEEE International Conference on Robotics and automation (ICRA), IEEE, pp 4628–4635
Nahangi M, Heins A, McCabe B, Schoellig A (2018) Automated localization of uavs in gps-denied indoor construction environments using fiducial markers. In: ISARC. Proceedings of the International Symposium on Automation and Robotics in Construction, IAARC Publications, vol 35, pp 1–7
Palacz W, Ślusarczyk G, Strug B, Grabfska E (2019) Indoor robot navigation using graph models based on bim/ifc. In: International Conference on Artificial Intelligence and Soft Computing, Springer, pp 654–665
Sato A, Nakajima M, Kohfike N (2019) Rapid ble beacon localization with range-only elf-slam using beacon interval constraint. In: 2019 International Conference on Indoor Positioning and Indoor Navigation (IPIN), IEEE, pp 1–8
Siemiatkowska B, Harasymowicz-Boggio B, Przybylski M, Różańska-Walczuk M, Wiśniowski M, Kowalski M (2013) Bim based indoor navigation system of hermes mobile robot. In: Romany 19-Robot Design, Dynamics and Control, Springer, pp 375–382
Song S, Marks E (2019) Construction site path planning optimization through bim. In: Computing in Civil Engineering 2019: Visualization, Information Modeling, and Simulation, American Society of Civil Engineers Reston, VA, pp 369–376
Strug B, Ślusarczyk G (2017) Reasoning about accessibility for disabled using building graph models based on bim/ifc. Visualization in Engineering 5(1):1–12
Thrun S (2002) Probabilistic robotics. Communications of the ACM 45(3):52–57
Wang Q, Zuo W, Guo Z, Li Q, Mei T, Qiao S (2020) Bim voxelization method supporting cell-based creation of a path-planning environment. Journal of Construction Engineering and Management 146(7):04020080

Xiang Y, Fox D (2017) Da-rnn: Semantic mapping with data associated recurrent neural networks. arXiv preprint arXiv:170303098

Yin H, Xia W, Zhang Y, Shen L (2016) Uwb-based indoor high precision localization system with robust unscented kalman filter. In: 2016 IEEE International Conference on Communication Systems (ICCS), IEEE, pp 1–6