Automation in Construction 146 (2023) 104647

Contents lists available at ScienceDirect

Automation in Construction

journal homepage: www.elsevier.com/locate/autcon

BIM-based indoor mobile robot initialization for construction automation using object detection

Xinge Zhao, Chien Chern Cheah *

School of Electrical and Electronic Engineering, Nanyang Technological University, 50 Nanyang Ave, Singapore, 639798, Singapore

ARTICLE INFO

Keywords:
Mobile robot initialization
Construction robot
Building Information Model (BIM)
Object Detection

ABSTRACT

In recent years, there has been increasing interest in robotic solutions to revolutionize the conventional construction industry. Despite various advances in developing mobile robotic solutions for construction automation. One key bottleneck towards a fully automated robotic solution in construction is the initialization of the mobile robot. Currently, most of the commercialized mobile construction robots are manually initialized before autonomous navigation can be performed at the construction sites for automated tasks. Even if the robot is initialized, the location information can be lost while navigating and re-initialization is required to resume the navigation. Any wrong initialization can cause failure in robot pose tracking and thus prevent the robot from performing the planned tasks. However, in indoor construction sites, GPS is not accessible, and indoor infrastructures, such as beacon devices are not available for robot initialization. In addition, construction environments are dynamic with significant change in scenes and structures for different construction blocks and floors, making pre-scanning of the environments and map matching difficult and time-consuming. An infrastructure-free and environment-independent robot initialization method is therefore required. In this paper, we propose an integrated Building Information Model (BIM)-based indoor robot initialization system using an object detector to automatically initialize the mobile robot when it is deployed at an unknown location. Convolutional neural network (CNN)-based object detection technique is used to detect and locate the visual features, which are widely distributed building components at construction sites. A feature matching algorithm is developed to correlate the acquired online information of detected features with geometric and semantic information retrieved from BIM. The robot location in the BIM coordinate frame is then estimated based on the feature association. Moreover, the proposed system aggregates the BIM information and the sensory information to supervise the online robot decision making, making the entire system fully automatic. The proposed system is validated through experiments in various environments including a university building and ongoing construction sites.

1. Introduction

The construction industry plays a pivotal role in the economic growth of many countries. Construction works are inherently labor-intensive, inefficient, and rely heavily on manual processes, which are prone to instrumental and human errors, and fatigue [1]. In recent years, there has been a rapid development of robotic solutions for construction automation that brings tremendous improvements in productivity, quality, and other aspects of construction work [2]. Various mobile robotic and mechatronic systems were developed to perform certain tasks in construction automation through navigating around the construction site. For example, a waste recycling robot was developed to inspect the entire construction environment and detect the nails and screw waste [3]. The construction monitoring robots mounted with scanning sensors and cameras were developed to navigate in the working environment to automate the construction inspection either for in-progress monitoring [4] or post-construction quality and defects check [5].

In the robot navigation stack, the robot initialization process is always the crucial component. As the robot is deployed at an unknown location in the working space, the robot's initial position in the map must first be determined so that the desired path to the target locations can be planned. In addition, if the robot loses its location while navigating, re-initialization is required to rectify its position on the map to resume the navigation. Any wrong initialization can cause failure in robot pose tracking and thus prevent the robot from performing the planned tasks. In indoor or GPS-challenged construction environments where spaces are concealed by building structures, acquisition of the

* Corresponding author.
E-mail addresses: xinge003@e.ntu.edu.sg (X. Zhao), ECCCheah@ntu.edu.sg (C.C. Cheah).

https://doi.org/10.1016/j.autcon.2022.104647
Received 29 August 2022; Received in revised form 16 October 2022; Accepted 29 October 2022
Available online 28 November 2022

location information is challenging. For most of the commercial products of indoor mobile robots, the robot initialization is performed manually by the robot operator. Such initialization is usually performed by expert users, such as robot engineers since it requires certain knowledge of the robot navigation system. To achieve fully automated robot navigation in construction sites, an autonomous robot initialization system is needed to make the system easily operated by non-expert users from any position. Some automated indoor positioning systems were proposed to realize the localization of mobile devices based on various technologies, such as Wi-Fi [6], Bluetooth Low Energy (BLE) [7], image processing [8] and laser scan matching [9]. Despite the extensive research in indoor positioning, the existing approaches are infrastructure-based or environment-dependent. Some methods require the physical deployment of instruments for localization. Infrastructure-free methods usually rely on pre-scanning and mapping of each different environment. The change in scenes and structures for various construction blocks at different construction stages makes it difficult and inefficient to apply such methods. Moreover, the initialization performance is usually poor for highly repetitive environments. Therefore, robot initialization remains a challenge for construction sites where telecommunication infrastructure is not yet available, operating environments are highly repetitive and symmetric, and robots are expected to work in multiple environments, such as different construction blocks.

This paper proposes an infrastructure-free and environment-independent BIM-based mobile robot initialization system to localize the robot in a known map built from BIM. CNN-based object detection is used to recognize visual features while geometric and semantic BIM information are utilized to associate with the sensory information to facilitate the robot location estimation. The proposed method provides a viable solution to realize the automated robot initialization in various construction environments without any requirement for pre-scanning or infrastructure deployment.

The main contributions of this paper are summarized below:

(a) This work utilizes a BIM-based feature association approach and CNN-based real-time object detection to realize the robot self-initialization. The proposed approach is infrastructure-free and therefore, can eliminate the human effort for robot initialization.

(b) The proposed method is environment-independent. It requires no prior mapping or scanning of the entire environment when applied to new environments with different scenes and layouts. The BIM information retrieval is standardized while the object detector can be employed for different environments that exhibit similar visual features, such as different blocks at the same construction site. It improves the robot initialization efficiency when the robot is required to work in multiple environments.

(c) This study illustrates the superiority of using BIM in robot initialization in various aspects. The semantic and geometric BIM information is not only used for the featuring matching and location estimation, but also utilized for the robot decision making and providing geometric constraints for localization. Therefore, the leverage of BIM can expedite the initialization process and improve the localization accuracy.

(d) The online robot decision making and active exploration approaches are proposed by fusing the BIM information and the sensory data, making the initialization automatic and require no manual control on the robot, and hence contribute towards the fully automated robot navigation system. Moreover, the proposed method includes the multi-level rectification of the robot's final pose that ensures accurate initialization results for robot navigation.

2. Related work

The positioning and localization for indoor mobile applications is a fundamental problem to carry out location-based services, such as indoor navigation, time-critical rescue, and all robot-related applications. Radio-frequency (RF) based methods using dedicated wireless communication networks were extensively studied. However, they were infrastructure-dependent due to the necessity for the deployment of signal transmitters and beacons. With the advances and ubiquitous availability of inertial sensors, cameras, and LiDAR, infrastructure-free methods using visual markers and simultaneous localization and mapping (SLAM) technology were developed. However, pre-scanning and pre-configuration for each different environment was usually performed with these methods, making them time-consuming. BIM that contained substantial information about the environments was therefore explored to integrate with the existing localization methods to either improve the localization accuracy or reduce the effort for environment configuration in construction automation. In this section, the infrastructure-dependent methods, infrastructure-free methods, and BIM-based localization are reviewed and discussed in detail.

2.1. *Infrastructure-dependent methods*

RF-based methods estimated the position by using the mobile device and the deployed infrastructure as a signal transmitter and receiver pair. The location was determined based on the measured distances and angles to the infrastructure with the known locations [10], or the measured signal strength pattern [11]. A wireless fidelity (WiFi)-based indoor positioning system was developed for labor tracking at a shield tunnel construction site [12]. Other methods used ultra-wideband (UWB) beacons [13] and radio frequency identification (RFID) [14] tags to integrate with robot motion data for positioning. However, these RF-based methods are not suitable for construction environments since the communication devices are not yet available and the deployment of infrastructure is time-consuming and not cost-effective. In addition, the movement of the workers and construction materials can cause instability in received signal strength and introduce severe multipath problems that impair localization accuracy.

2.2. *Infrastructure-free methods*

Infrastructure-free localization methods were developed using visual and laser sensor data to eliminate the requirement for deployment of signal transmitters and beacons. Fiducial markers, real visual features in images, and laser scans were used to recognize locations through feature matching and scene understanding.

2.2.1. *Marker-based methods*

One of the alternatives for accurate and economical localization was to deploy artificial markers such as QR codes [15] and ArUco tags [16] with encoded location information. QR codes were used as global reference points for a mobile robot localization system [17] with the markers deployed on the ceilings. A more generalized and real-time QR code-based method was proposed in [18] to estimate the 3D pose of the camera with varying orientations based on extended image processing. However, marker-based methods still require substantial effort to configure the environment. When a robot is deployed to detect markers through self-exploration, markers can be easily missed or occluded by obstacles. Moreover, the markers are susceptible to wear and tear during the construction work, and thus result in localization failure.

2.2.2. *SLAM-based methods*

In indoor GPS-challenged environments, simultaneous localization and mapping (SLAM) technology is usually used for mobile robots to generate a map of the unknown environment while localizing the robots in the constructed map without any infrastructure deployment. Visual SLAM (vSLAM) was extensively studied for cost-effective indoor localization. Mono-SLAM [19] estimated the pose state through matching of image patches and optimized the estimation using extended Kalman filter (EKF). Klein and Murray [20] introduced the concept of keyframes and proposed a framework of parallel tracking and mapping (PTAM). Improved from PTAM, another keyframe-based system, ORB-SLAM [21] was developed for tracking, mapping, relocalization, and loop closing, and obtained high popularity in commercialized applications. It was extended to ORB-SLAM2 [22] for RGB-D cameras. OpenVSLAM [23] was proposed based on ORB-SLAM while improved for higher extensibility and accuracy. In the stack of vSLAM, relocalization was an essential module to determine the camera pose from the built map and recover from tracking failure using place recognition techniques. In construction automation, instead of the location in the vSLAM map, the robot's initial position in a reference coordinate system was expected to navigate to a target location with known positions in the same coordinate system. In an in-building emergency response system, Po-Yen [24] marked the fixed target positions in the built feature map from vSLAM to link the initial position and destinations. However, accurate mapping using vSLAM with exact matching to the physical world was challenging for dynamic construction environments with unstructured regions, temporarily placed materials, and presence of workers. Alternatively, the geo-tagged database of images or features was constructed to directly match the query image to the reference image with the known absolute location in a global coordinate system using visual features [25,26] for localization. The recent success of deep neural networks on a variety of domains has boosted research on vision-based localization using convolutional neural network (CNN) for more robust image retrieval. In [27], Mask R-CNN network was used to detect objects in the image, and SURF descriptors [28] were applied to each object to retrieve object information for keyframe matching. In [29], VGG16 based image descriptors were applied to extract Top-N similar images, and Faster RCNN based region descriptors were leveraged to re-rank the image candidates for robust image retrieval. Although these works achieved infrastructure-free localization, it required time-consuming pre-mapping of the whole environment to build a high-resolution geo-tagged database of visual features for accurate localization. Moreover, the rapid variations of scenes due to ongoing construction work can easily cause failure in detecting corresponding keyframes and a frequent construction of the visual feature database to overcome such a problem is inefficient.

Besides vSLAM, laser-based SLAM algorithms with higher accuracy and faster computation were also explored for robot localization. Gmapping [30] was a widely used SLAM algorithm based on the improved Rao-Blackwellized Particle Filter (RBPF). Another efficient 2D SLAM algorithm, Hector SLAM [31] used a bilinear filtering function to estimate grid occupancy and Gauss Newton method to optimize scan matching for a real-time localization. However, these Bayes-based methods lacked loop closure detection and optimization and therefore, were not capable of eliminating accumulated localization errors. The graph-based SLAM, such as Karto SLAM, and Cartographer [32], in the contrast, used observations as constraints and modeled a graph-based representation for global optimization which minimized accumulated drift errors. The Cartographer algorithm that included local scan matching, back-end loop closure detection, and subgraph optimization achieved high accuracy and robust real-time performance. The location tracking methods in laser-based SLAM could also be used to globally localize the robot from an unknown initial position in a given map. Scan matching algorithms [9,33] were applied to match the query scan to a database of keypoints generated from the grid-based maps and retrieve the associated location information. Other probabilistic localization algorithms [34–37] were variants of the Bayes filter which transformed the probabilistic belief of robot pose with robot motion and observation models. However, the prior mapping of the entire environment was required with these methods, and the discrepancies between the built map and the physical environment resulted in inaccurate global localization. The AMCL software [38] offered a convenient registration between a 2D ground truth map, such as a designed floor plan of a building, to the robot laser scans for fine-grained global localization. Although it performed well for environments with rich geometric features, it was vulnerable to wrong localization in repetitive environments [39]. Therefore, it was not feasible for large-scale complex construction environments with many similar plain walls, pillars and long corridors.

2.3. BIM-based localization

Many research has studied the potential of BIM in robot path planning [40] and position tracking [41] using the indoor mobility information in BIM, such as the size of areas and the accessibility to other spaces and transitions [42]. BIM was also used in infrastructure-dependent indoor localization systems to provide the environment information. A hybrid asset tracking system for location awareness in indoor construction environments was developed integrating the Bluetooth Low Energy (BLE) technology, motion sensors and BIM [43]. In this application, the information of building components from BIM provided additional knowledge of the geometric boundaries and inaccessible locations, which were used to improve the accuracy of position tracking. In [44], a BIM-based indoor localization algorithm was proposed to assist the building fire emergency response operations. Algorithms for beacon deployment leveraging BIM were developed and the proposed localization schema helped improve the accuracy of room-level localization. However, the aforementioned methods still relied on traditional RF-based localization approaches and hence, required the device deployment.

Other methods fused BIM information with visual and LiDAR observations to eliminate online surveying operation in traditional SLAM-based methods. VGG network was implemented to extract image features and match captured real images with a configured geo-tagged database of rendered BIM images [45]. However, it only determined the matched image while the exact camera pose was not estimated. Debaditya et al. [46] developed a camera pose regressor based on the PoseNet architecture with synthetic training images rendered from a 3D BIM model. However, the error of 2 meters for localization with this method was insufficient for accurate robot navigation. Although the time taken for on-site image collection was reduced in the aforementioned methods, additional offline effort to construct a geo-tagged dataset for a large number of rendered BIM images was still time-consuming. As the layouts varied for different blocks, a separate database of BIM images is required for each different environment, making it environment-dependent and less efficient. In [47], a BIM-based interface was proposed to convert a BIM model into serialized state representations of a 3D SLAM. It localized the robot by matching the online SLAM with the composed mapping state. This method could be used to recognize the robot's location without surveying the entire working space. However, it still required the robot to start from a pre-mapped zone where the region around the initial position should be sufficiently similar to the reference model. Therefore, it is not suitable for highly repetitive and dynamic construction environments, with rooms of the same size, long corridors without closed boundaries in the laser-scans, and randomly placed construction materials that introduced uncertainty in scan matching. Such characteristics could result in poor initialization due to the lack of a unique matched scan.

Based on the review, existing methods require infrastructure deployment or prior scanning and configuration for each different environment, making them difficult and inefficient to be deployed for a fully automated robot initialization. In addition, the characteristics of construction sites, such as the high repetitiveness of environments, and presence of workers and construction materials are not well considered. Therefore, an autonomous robot initialization system for construction automation is required.

3. BIM-based mobile robot initialization system using object detection

In this paper, a BIM-based robot initialization system which is infrastructure-free and environment-independent is proposed to achieve autonomous robot initialization, and contribute as a component towards fully automated robot navigation for construction automation without any pre-scanning of the environment.

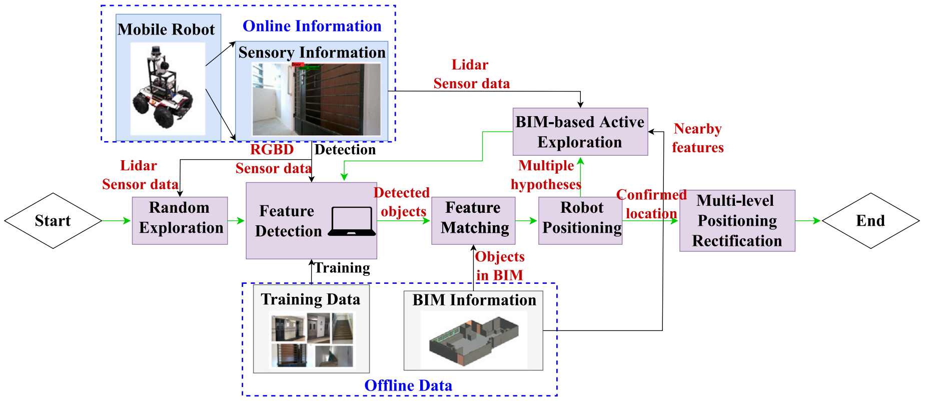


Fig. 1. BIM-based mobile robot initialization system.

3.1. System overview

The overall initialization system and information flows among the sub-systems are illustrated in Fig. 1. The offline image database is built and utilized to train an object detector for building components, such as doors, lifts and staircases. Both semantic and geometric information of the building components is retrieved from BIM to provide knowledge of the environment, and each component is considered as a feature.

In the online phase, as the initialization starts from an unknown location in the operating environment, without any knowledge of the surrounding environment, the robot conducts random exploration to search for visual features using the trained object detector. The vision system carries out object detection, processes detection results, and sends signals to the navigation system for decision making of robot actions, such as stopping of the robot, and recording of feature information once a new feature is detected by the vision system. The feature matching program takes BIM information and online sensory information to determine the association between detected features and the BIM feature list. It evaluates the confidence level in which the robot locates at a particular location based on the matching outcomes. If there are multiple hypotheses of the robot position, active exploration using BIM information is triggered to make the robot find nearby features and validate the most probable hypothesis. As more features are detected through random exploration and the active exploration, the false hypotheses are opted out and the unique match between BIM and the online sensory information can be determined. It marks the success of coarse initialization by identifying the rough region where the robot locates. Then the multi-level positioning rectification is applied to fine tune the initialization result and improve the localization accuracy.

3.2. Offline phase

3.2.1. Feature information retrieval from BIM

In the proposed robot initialization system, BIM is converted to the IFC file format and then exported into the Autodesk Revit software. The information retrieval approach is developed based on the built-in Dynamo software in Revit. An application is created to extract information from BIM and export it to a text file. The user may select the elements by clicking elements from the graphical user interface of BIM software or simply specifying the category of interest (i.e. doors, lifts, etc.). The selected elements are then passed to a customized program that retrieved the semantic and geometric attributes of the building components. The retrieved semantic information includes the unique element ID number, and the category of elements while geometric information contains the element locations in the 2D floor plan, the normal vector to the front surface of the structure component, and the dimensions of the element, as shown in Table 1.

Table 1
BIM information for building features.

ID	Category	X (m)	Y (m)	Norm_X	Norm_Y	Width (m)	Height (m)
62938	Doors	−6.38	−4.14	1	0	1.05	2.30
96688	Lift	−5.07	0.00	0	−1	1.10	2.12
94617	Stairs	−3.39	58.41	−1	0	1.19	1.86

3.2.2. Training of CNN-based object detector

CNN-based improved version of You Only Look Once (YOLOv3) [48] architecture is the backbone framework for the object detection system in the proposed method. Since real-time object detection is required to carry out the online initialization, YOLOv3 [48] is chosen for its good performance in real-time applications and high detection accuracy as well. However, the initialization system is modular and the YOLOv3 architecture can be replaced by other real-time object detection frameworks, such as YOLOv4 [49], YOLOR [50], and Mobilenet-SSDv2 [51].

Based on the YOLOv3 framework, the detection model is trained to recognize common architectural components at the construction sites, such as doors, lifts and staircases. The convolutional weights that are pre-trained on ImageNet using Darknet-53 are set as the initial weights. The trained image dataset is composed of images extracted from internet resources and captured at university buildings and construction sites. Images taken at construction sites exhibit diversity in lighting conditions, viewing angles and construction stages to ensure that the detector can be applied at varying construction stages and different construction sites. Some sample images collected from different resources are shown in Fig. 2(a). Training images are fed into the detector framework with a size of 64 images per batch and weight parameters are updated after each iteration of processing 16 images. A total of 4000 iterations are performed to obtain the final detection model. The trained model is used to detect and localize the building components. Some sample detection results are presented in Fig. 2(b).

3.3. Online robot initialization

The trained real-time object detector and the retrieved BIM information in the offline phase are then utilized in the online phase by integrating with the online sensory information to guide robot motion and locate the robot.

3.3.1. Detection and localization of features

The object detection and localization using the pre-trained YOLOv3 object detector and depth measurements with RGBD sensor is developed. Moreover, the techniques to process the results of object detection and recognize distinct features from the continuous image stream are included. With the implementation of the proposed techniques, the information of all the building components that have been

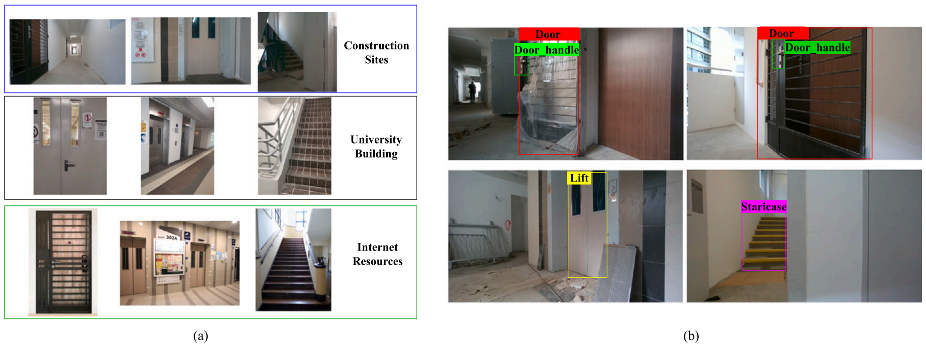


Fig. 2. (a) Image dataset for the training of object detector (b) Object detection results.

seen since the initialization starts can be obtained and recorded in a feature list.

As the robot explores and navigates in the working environment, the real-time object detector is executed continuously. In each raw image, if an object is detected, the 3D location of the object with respect to a fixed odometry frame is calculated. The odometry coordinate frame is established depending on the starting robot pose where the initialization begins. The origin of the odometry frame is the starting robot position and the positive x-direction of the odometry frame is the robot facing direction.

To localize an object, camera intrinsic parameters, 2D object location in the image frame and depth to the object are required. The camera calibration is performed, and intrinsic parameters including image center point (p_x, p_y), focal length (f), and distortion coefficients are obtained. From the object detection results, the object center (b_x, b_y) in the image frame can be obtained. The mean value of the depth measurements acquired from an RGBD sensor in a $(2n+1)\times(2n+1)$ pixels square region around this object center, denoted as D is used to localize the object. The object position in the 2D image frame is transformed to the 3D point in the camera frame as follows:

$$\begin{pmatrix} x_c \\ y_c \\ z_c \end{pmatrix} = \begin{pmatrix} (b_x - p_x) \times \dfrac{D}{f} \\ (b_y - p_y) \times \dfrac{D}{f} \\ D \end{pmatrix} \tag{1}$$

where x_c, y_c, z_c represent the coordinates of the detected object with respect to the camera frame. The object position in the camera frame is further converted to the position in the odometry frame through successive homogeneous transformations as follows:

$$\begin{pmatrix} x_o \\ y_o \\ z_o \\ 1 \end{pmatrix} = T^{robot}_{odom} \cdot T^{lidar}_{robot} \cdot T^{camera}_{lidar} \begin{pmatrix} x_c \\ y_c \\ z_c \\ 1 \end{pmatrix} \tag{2}$$

where $T^{f_1}_{f_2}$ is the homogeneous transformation matrix from one frame f_1 to the other frame f_2 and x_o, y_o, z_o are the coordinates of the detected object in the odometry frame. Given the aforementioned equations, each detected object can be localized to a 3D position in the odometry frame.

For each raw image, a set which contains the detected objects in the current image and the information of each object, including the class and the position of the object in the odometry frame can be obtained. The information from individual raw images is aggregated and processed to acquire the information of all the observed features since the initialization starts using a detection filter. The set of all the features that have been detected until the current image, S_{object} with a total of n^S features is denoted as:

$$S_{object} = (s_1, s_2, s_3 \dots s_{n^S}) \tag{3}$$

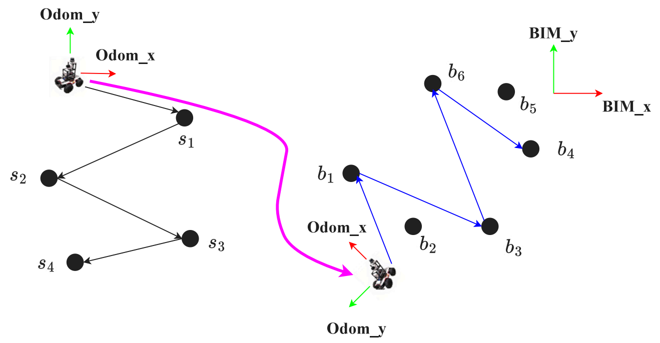


Fig. 3. Matched features in different coordinate frames.

3.3.2. Feature matching

Similarly, the set of BIM features retrieved from BIM in the offline phase, B_{object} with a total of n^B features is denoted as:

$$B_{object} = (b_1, b_2, b_3 \dots b_{n^B}) \tag{4}$$

A feature matching algorithm is developed to map each detected feature in S_{object} to a feature in B_{object} (BIM) using semantic information and geometric relationships between features. For each detected feature in S_{object}, there exists an identical feature in BIM that matches with it but the features are defined in different coordinates since the locations of features in S_{object} and in B_{object} are with respect to the odometry frame and BIM frame respectively. An example is shown in Fig. 3, given the sequence of the features detected as (s_1, s_2, s_3, s_4) and six BIM features in the BIM list as $(b_1, b_2, b_3, b_4, b_5, b_6)$. A feasible match for this example is denoted as $(s_1 : b_1, s_2 : b_3, s_3 : b_6, s_4 : b_4)$, representing each pair of detected feature and the corresponding matched BIM feature. Since there is an unknown transformation between the odometry frame and the BIM frame, absolute feature coordinate information cannot be used to match the two sets of features but the relative positions among the features are leveraged.

For any pair of BIM features b_u and $b_v \in B_{object}$, as defined in (4) with x_u, x_v, and y_u, y_v denote the x and y coordinates for the two features respectively, the distance between them in the BIM frame is represented by d^B_{uv} as:

$$d^B_{uv} = \sqrt{(x_v - x_u)^2 + (y_v - y_u)^2} \tag{5}$$

Similarly, for any pair of detected features s_i and $s_j \in S_{object}$, the distance in the odometry frame is denoted as d^S_{ij}. For the same pair of BIM features b_u and b_v, the relative direction vector in the BIM frame $(v^B_{uv,x}, v^B_{uv,y})$ is defined as:

$$(v^B_{uv,x}, v^B_{uv,y}) = \left(\frac{x_v - x_u}{d^B_{uv}}, \frac{y_v - y_u}{d^B_{uv}}\right) \tag{6}$$

and for any pair of detected features s_i and $s_j \in S_{object}$, the relative direction vector in the odometry frame is denoted as $(v^S_{ij,x}, v^S_{ij,y})$. Since the relative positions between features are used to determine the match, the distances and relative directions for all the pairs of features are calculated for both feature sets. Given any two detected features (s_i, s_j) and any two BIM features (b_u, b_v), the feature labels and the distance between the features are used as the criteria to compare the feature patterns. To match two features, it should be ensured that the feature classes are matched such that $label(s_i) = label(b_u)$ and $label(s_j) = label(b_v)$, and the distance between features (s_i, s_j) matches with the distance between features (b_u, b_v). To accommodate localization errors caused by the fluctuation of sensor measurements, a match is taken as feasible when the distance error between the BIM features and detected features is within a predefined threshold. A valid match is defined with a function z_d where $z_d(\{s_i, b_u\}, \{s_j, b_v\}) = 1$ represents a valid match. The function for two feature matching is defined as:

$$z_d(\{s_i, b_u\}, \{s_j, b_v\}) = \begin{cases} 1, & \text{if } |d^B_{uv} - d^S_{ij}| \leq \alpha \\ 0, & \text{otherwise} \end{cases} \tag{7}$$

where α is the threshold for the error in distance. The criteria to match a pair of features can be extended to match a feature pattern with three features. To match a sequence of detected features (s_i, s_j, s_k) to a sequence of BIM features (b_u, b_v, b_w), the triangular pattern that is formed by the detected features should be matched to a corresponding feature pattern in BIM. Besides the criteria for label matching and distance matching, angle matching is also considered since the sequence of the detected features should be considered to opt-out false matches. For example, the matched BIM feature sequences (b_u, b_v, b_w) and (b_u, b_w, b_v) have the same feature pattern geometrically as shown in Fig. 4, but represent different travel paths, leading to one estimation that the robot starts at b_u and currently locates at b_w while the other estimation that the path ends at b_v. Given the rotation angle from the first direction vector, v^S_{ij} to the second direction vector, v^S_{ik} is anti-clockwise for the detected features, the matched BIM feature sequence can only be (b_u, b_v, b_w). For a group of three features (b_u, b_v, b_w) the rotation angle (θ_{uvw}) from direction vector v^B_{uv} to v^B_{uw} is defined as:

$$\theta_{uvw} = atan2\left(\frac{v^B_{uv,x} v^B_{uw,y} - v^B_{uv,y} v^B_{uw,x}}{v^B_{uv,x} v^B_{uw,x} + v^B_{uv,y} v^B_{uw,y}}\right) \tag{8}$$

Similarly, the rotation angle from v^S_{ij} to v^S_{ik} for detected features (s_i, s_j, s_k) in the odometry frame is calculated and represented by θ_{ijk}. Therefore, the criteria for the matching of three features is defined by integrating the criteria for two features matching and the requirements for the conformance of rotation angles. Next, $z(\{s_i, b_u\}, \{s_j, b_v\}, \{s_k, b_w\}) = 1$ represents a feasible match and z is defined as:

$$z(\{s_i, b_u\}, \{s_j, b_v\}, \{s_k, b_w\}) = \begin{cases} 1, & \text{if } z_d(\{s_i, b_u\}, \{s_j, b_v\}) = 1, \\ & z_d(\{s_i, b_u\}, \{s_k, b_w\}) = 1, \\ & |\theta_{uvw} - \theta_{ijk}| \leq \beta \\ 0, & \text{otherwise} \end{cases} \tag{9}$$

where β is the threshold for the difference in the measurement of angle. The approach to match three features is then extended to determine a complete match $m : S_{object} \rightarrow B_{object}$ that matches each element $s_i (i = 1, 2, 3, \ldots, n^S)$ in S_{object} to a corresponding BIM feature denoted as $m(s_i)$, The pseudocode of the feature matching algorithm is given in Algorithm 1.

The matching process is performed using a bottom-up approach. Given the first feature in S_{object} to be s_1, all the features in the BIM list that have the same label as s_1 are selected as possible candidates. Based on the possible matches for the first feature, it is then analyzed to check if any remaining feature in the BIM can match with the second detected feature s_2 using (7). Then with all the feasible matches for the s_1 and s_2, each combination is evaluated to add the third feature that satisfies the match criteria given in (9). For all the following features in S_{object},

Algorithm 1: Feature matching algorithm

```
    Input: BIM feature set ($B_{object}$), Detected feature set ($S_{object}$)
    Output: The set of feasible matches ($M$)
    Initialize: $M = \emptyset$
 1  for each $b_u \in B$ and label($b_u$)=label($s_1$) do
 2      $M_u \leftarrow \emptyset$;
 3      for each $b_v \in B_{object}$ and $b_v \neq b_u$ do
 4          $M_u \leftarrow M_u \cup [(s_1, b_u), (s_2, b_v)]$ if label($b_v$)=label($s_2$) and
              $z(\{s_1, b_u\}, \{s_2, b_v\}) = 1$;
 5      end
 6      for each $s_i \in S_{object}$ that $i \geq 3$ do
 7          $E \leftarrow \emptyset$;
 8          for each $m \in M_u$ and $b_w \in B_{object}$ and w not matched with any
              feature in $S_{object}$ do
 9              $m \leftarrow m \cup (s_i, b_w)$ if label($b_w$)=label($s_i$) and
10               $z(\{s_{i-2}, m(s_{i-2})\}, \{s_{i-1}, m(s_{i-1})\}, \{s_i, b_w\}) = 1$ ;
11              $E \leftarrow E \cup m$ ;
12          end
13          $M_u \leftarrow E$;
14      end
15      $M \leftarrow M \cup M_u$;
16  end
```

it forms a group of three features with the previous two features and the matching criteria for three features matching is applied to obtain the BIM feature that matches with the newly detected feature. The iteration is repeated until all the features in S_{object} are matched with a feature in B_{object}. The matches in the last iteration are removed if it cannot find any feature in BIM to match the coming feature in the new iteration. Note that there can be many possible matches with fewer features present, but many false candidates can be discarded as new features are added in the bottom-up matching process.

3.3.3. Robot positioning and BIM-based active exploration

Since similar geometric patterns can occur at different locations, and thresholds are set for feature matching to accommodate localization errors, there can be multiple matches in M that satisfy the requirements to match a detected feature list S_{object} with the BIM feature list B_{object}. Each possible match is associated with a hypothesis of the path and the current robot position. Based on the location information of the last three detected features, the current robot position can be estimated through the principle of trilateration, as illustrated in Fig. 5(a). Let the coordinates of the features be (x_i, y_i) and the distance to the feature be d_i, where $i = 1,2,3$ for the last three detected features, the robot location (x_g, y_g) is determined to minimize the sum of squares between the measured distances to the features and the calculated distances based on the estimated position as:

$$(x_g, y_g) = \arg\min_{x,y} \sum_{i=1}^{3} \|{d_i}^2 - ((x - x_i)^2 + (y - y_i)^2)\|^2 \tag{10}$$

Eq. (10) can be solved with the nonlinear least square optimization method [52].

As the robot position in the BIM frame, (x_g, y_g) is determined, the robot heading direction can then be estimated based on the transformations between the robot local frame and the global BIM frame, and the location information of the last detected feature that is in the current robot's view, as shown in Fig. 5(b). Suppose the robot yaw angle is θ_g in the BIM frame and hence, the translation and rotation from BIM frame to the robot frame are x_g, y_g, θ_g, respectively. Let the last feature location in the robot frame be (o_{rx}, o_{ry}), the coordinates of the last detected feature in the BIM frame (o_{bx}, o_{by}) can be expressed as:

$$\begin{bmatrix} o_{bx} \\ o_{by} \\ 1 \end{bmatrix} = \begin{bmatrix} \cos\theta_g & -\sin\theta_g & x_g \\ \sin\theta_g & \cos\theta_g & y_g \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} o_{rx} \\ o_{ry} \\ 1 \end{bmatrix} \tag{11}$$

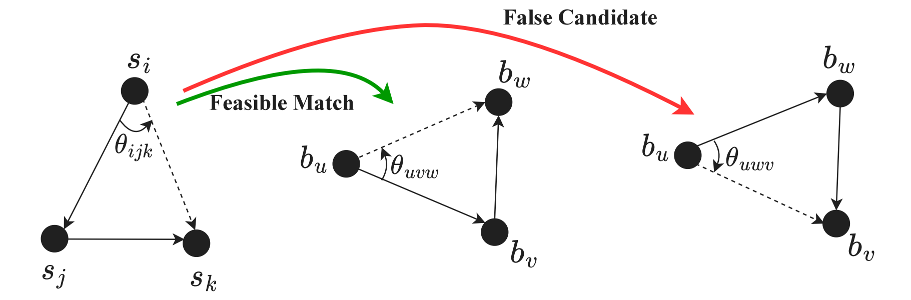


Fig. 4. Matching of three features.

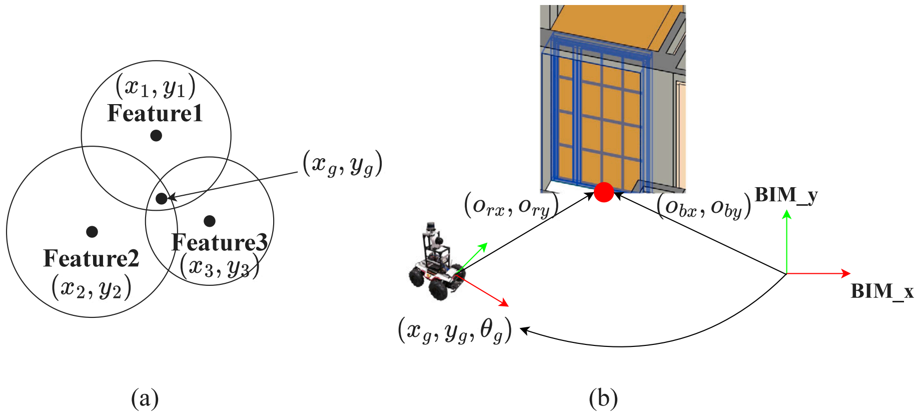


Fig. 5. Robot localization (a) positioning using trilateration (b) estimation of orientation.

Since the feature coordinates in the robot frame and the BIM frame are known and the robot position in the BIM frame (x_g, y_g) is determined, the equations in (11) can be rearranged taking $\cos\theta_g$ and $\sin\theta_g$ as the variables:

$$\begin{bmatrix} o_{rx} & -o_{ry} \\ o_{ry} & o_{rx} \end{bmatrix} \begin{bmatrix} \cos\theta_g \\ \sin\theta_g \end{bmatrix} = \begin{bmatrix} o_{bx} - x_g \\ o_{by} - y_g \end{bmatrix} \tag{12}$$

Then $\cos\theta_g$ and $\sin\theta_g$ can be solved with the linear equations and θ_g can be determined as:

$$\theta_g = atan2(\frac{\sin\theta_g}{\cos\theta_g}) = atan2(\frac{(o_{by} - y_g)o_{rx} - (o_{bx} - x_g)o_{ry}}{(o_{bx} - x_g)o_{rx} + (o_{by} - y_g)o_{ry}}) \tag{13}$$

Therefore, for each feasible match, the corresponding current robot pose, (x_g, y_g, θ_g) can be calculated. However, the hypotheses of robot locations are not equally probable, with the probabilities higher at selected locations where the detected feature pattern exhibits a higher similarity with the BIM feature pattern. BIM-based active exploration is then introduced to accelerate the initialization process by making the robot navigate based on the most probable estimation and try to find more distinct features in the neighborhood to increase the confidence level of the estimation. It is an active exploration step to exploit the online detection information and BIM information to validate the hypothesis of the robot position and quickly opt-out false matches, instead of spending more time exploring the environment randomly.

To facilitate the BIM-based active exploration, possible matches are sorted firstly based on the similarity between the detected feature pattern and the BIM feature pattern, which is quantified by the matching cost that measures the differences between feature patterns. Based on the criteria for feature matching introduced in (7) and (9), the cost for the match can be defined analogically. The total cost of a match m for the detected feature list S_{object} that incorporates the matching cost for the first two features, and the cost for all the following groups of three consecutive features is calculated as:

$$J = |d^B_{m(s_1)m(s_2)} - d^S_{s_1 s_2}| + \sum_{i=3}^{n^S} (|d^B_{m(s_{i-2})m(s_{i-1})} - d^S_{s_{i-2}s_{i-1}}| + |d^B_{m(s_{i-2})m(s_i)} - d^S_{s_{i-2}s_i}| + |\theta_{m(s_{i-2})m(s_{i-1})m(s_i)} - \theta_{s_{i-2}s_{i-1}s_i}|) \tag{14}$$

Once the most probable match that yields the least matching cost is determined using (14), the most probable robot position can be estimated subsequently. Based on the BIM information, the feature that is closest to the robot in the category with the highest priority is selected as the target object for active exploration. A category that contains fewer features exhibits a higher priority since the features are more distinct and more false matches can be removed if the feature is detected. To navigate to the target location, a modified local navigation algorithm based on bug algorithm [53] is developed. The Bug algorithm is selected since it works for the situation where the locations and shapes of obstacles in their environment are unknown but the target's relative position is available. The proposed modified algorithm is developed to make the navigation more efficient considering the characteristics of indoor construction environments with long corridors and the lift lobby. An example that illustrates the robot local navigation algorithm is shown in Fig. 6(a).

The distances to nearest obstacles in the front, left, right, front left, and front right regions of the robot are acquired from the laser scan data to facilitate the active exploration. Firstly, the robot rotates to the direction that points to the target feature location and then navigates to the target point. Given the location of the target feature to be (x_t, y_t), the desired yaw angle yaw_d and yaw error Δ_{yaw} are calculated where (x, y, yaw) is the robot pose at the instant time.

$$yaw_d = atan2(\frac{y_t - y}{x_t - x}) \tag{15}$$

$$\Delta_{yaw} = yaw_d - yaw \tag{16}$$

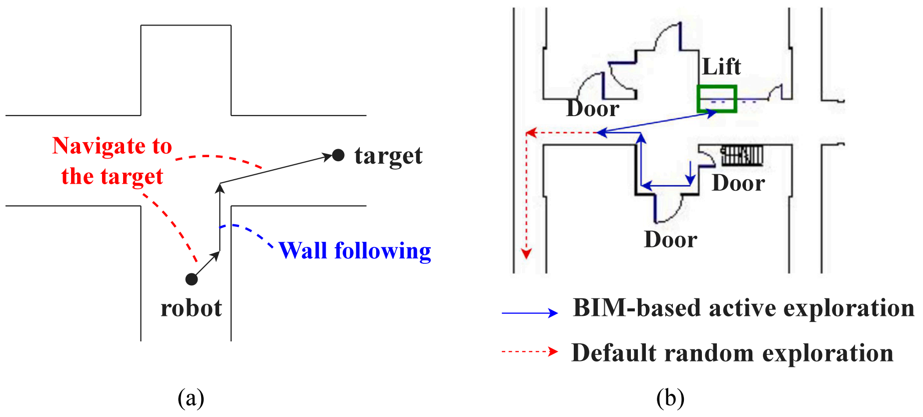


Fig. 6. (a) Robot local navigation (b) Application of the BIM-based active exploration.

It goes straight while the yaw error is within a predefined threshold, and adjusts the yaw angle if it deviates from the planned straight path connecting the robot pose and the target location. Once an obstacle is observed in the front, the robot avoids the obstacle by switching to the status of wall following. Depending on the orientation of the robot to the facing wall, either the left wall following or the right wall following is performed to reduce the distance between the robot and the target point. On the way the robot follows the wall, once reaching the end of a wall and meeting the condition that there is no obstacle between the robot and the target, the robot state is switched from wall following to navigating to the target point.

The implementation of BIM-based active exploration to locally tune the robot path and hence accelerate the initialization process is illustrated in Fig. 6(b). After three doors are detected with the random exploration, the match with the least cost is determined and the active exploration is executed to validate the hypothesis. Since there are few lifts present in the floor plan and one is supposed to be nearby based on the BIM information, the robot attempts to approach the nearby lift instead of continuing to randomly explore the environment. As the lift is detected and this more distinctive feature is added to the feature list, the wrong matching candidates are opted out quickly as a lift is not observable for most of the other matches.

3.3.4. Online decision making for coarse initialization

Based on the components of the online initialization phase introduced in the above sections, the robot continues to explore the environment, detects features, matches the features and executes active exploration when necessary until the only possible match is found.

The online decision making process takes BIM information and sensor data as input and controls the robot motion and the execution of programs in sub-systems. When the robot is deployed at an unknown location in the operating environment, the wall following algorithm is applied to facilitate the random exploration. By default, the robot follows the left wall and keeps going straight. It turns right at an inner corner, where there is an obstacle in the front and it turns left to find another piece of wall as it comes to an outer corner, where the wall it has followed comes to an end. As the robot explores the environment, the real-time object detection program is continuously executed at a frequency of 5 images per second. The filtering of detected objects in the image stream is applied to avoid false or duplicated recordings of the feature. Once three or more features are added to the confirmed feature set S_{object}, the association between detected features and features in BIM can be established and hypotheses of the robot pose are feasible. All the possible matches are passed to the list M and the corresponding estimated robot poses are determined. In the case that only one feasible match exists, the robot location can be determined directly and the initialization process is completed. While multiple matches are provided in M and the number of possible matches is lower than a threshold m_{thre}, the match with the highest confidence is adopted and the estimated robot location (x_g, y_g) is taken to validate the hypothesis. The nearest feature location (x_t, y_t) is determined using the robot location (x_g, y_g) and BIM information. In the process of the active exploration to the target point (x_t, y_t), if a new object is detected, the detected feature set S_{object} is updated and the new target for active exploration is generated according to the latest M. On the other hand, supposing no object is detected until a pre-defined time limit t_{thre}, the recovery behavior is carried out to extend the robot view to a wider angle so that missing detections that may be caused by view occlusion are avoided. Moreover, if there is no object detected even after the execution of recovery, it indicates that the target goal point may not be achievable and the prediction of robot pose is incorrect. Then the robot state is switched back to random exploration to find other features. The initialization process is terminated when the unique match is found. It marks the end of the coarse localization where the robot moving route and rough robot position are obtained using (10) and (13). The online decision making algorithm allows the robot to traverse throughout the environment automatically. Given the capability to switch between different states, such as stopping of the robot, recording of feature information, and navigating to nearby features based on the online sensor data and BIM information, the initialization is achieved without manual control.

3.3.5. Multi-level positioning rectification for fine initialization

Since there are inevitable errors in the measurements of robot odometry, noise in depth measurements and deviation between the bounding box centers and object centers, the coarse robot localization result that relies on the odometry data and object detection results may not be very accurate. Multi-level positioning rectification is used to further improve the robot positioning accuracy. As shown in Fig. 7(a), the raw trilateration result is first rectified using BIM information of geometric boundaries. The position inside the inaccessible regions is shifted to open areas. If a more accurate localization result is required, particle filter-based fine tuning can then be applied. The robot takes the rectified robot pose using BIM information as the initial estimation and positioning tracking is applied to fine tune the localization result until the covariance of robot pose distribution is lower than a predefined threshold.

BIM information for all the walls is retrieved to provide boundary information of the reachable areas. Based on the start position, end position and the surface norm of the walls, the perpendicular distances to the estimated robot pose are obtained. The nearest wall to the robot is then determined by sorting the distances. Since the robot follows the wall with a predefined threshold d_{thre}, any robot pose estimated with a smaller distance to the wall is unrealistic and geometric constraints

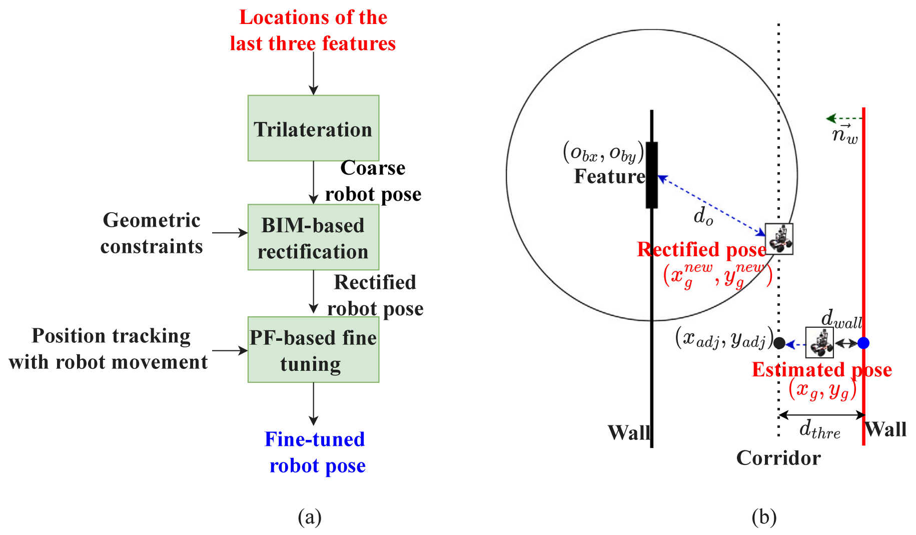


Fig. 7. (a) Sequence of multi-level positioning rectification (b) BIM-based rectification.

are applied to rectify the robot pose to keep away from the walls. An example is shown in Fig. 7(b), depicting the corridor and boundaries, the last detected feature, the estimated robot position and the rectified position. As the distance between the estimated robot position (x_g, y_g) and the wall is d_{wall}, which is smaller than the distance threshold, the estimated location should be adjusted to accessible regions first by keeping an allowable distance to the wall. The coordinates of the adjusted point (x_{adj}, y_{adj}) can be determined based on the ratio of d_{wall} over d_{thre}.

The rectified robot position is then determined based on the adjusted point while considering the distance to the last detected feature. It is ensured that the distance between the rectified position and the wall is the predefined threshold d_{thre} and the distance to the last detected feature is the measured distance d_o, while the point falls within the corridor. Let the wall surface norm retrieved from BIM be $\vec{n_w}$, and the BIM coordinates of the last detected feature be (o_{bx}, o_{by}), the rectified robot position (x_g^{new}, y_g^{new}) is determined as:

$$\begin{aligned}(x_g^{new}, y_g^{new}) &= \arg\min_{x,y}[(x-x_{adj})^2 + (y-y_{adj})^2] \ \textit{subject to} \\ \vec{n_w}\cdot(x-x_{adj}, y-y_{adj}) &= 0, \ \textit{and} \ (x-o_{bx})^2 + (y-o_{by})^2 = {d_o}^2\end{aligned} \tag{17}$$

The corresponding new robot heading θ_g^{new} can be determined with the detection results of the last feature using (13). The rectified robot pose using BIM information can be taken as the initial pose for autonomous robot navigation to other target locations directly. When there are obstacles or narrow passages, tolerance for the localization error must be smaller in order to avoid collisions. In-place rotation is usually performed to recover and adjust the localization results. However, it can easily fail if the robot cannot rotate for 360 degrees and eventually causes a navigation failure. To make the navigation smoother and localization more accurate, probabilistic-based methods can be applied to fine tune the pose estimation using the laser sensor observations before passing the initialization result to the navigation stack.

The ROS AMCL package [38] is used for the fine tuning of the pose estimation by applying robot position tracking. The problem is formulated in the way that the robot pose x_t at time step t is a random variable and $x_t = [x, y, \theta]^T \in \mathbb{R}^2 \times \mathbb{S}$. Taking the map m as input, the belief of the robot pose $bel(x_t)$ is updated recursively incorporating the motion model and observation model as [54]:

$$bel(x_t) = \eta \ p(z_t|x_t, m) \int p(x_t|x_{t-1}, u_t, m) bel(x_{t-1})\, dx \tag{18}$$

where η is a normalization constant, z_t and u_t are the observation and control inputs at time t.

The robot is controlled to follow the wall and move from the stopping position of coarse positioning for another few meters, given the output from BIM-based rectification as an input for the AMCL position tracking algorithm. The robot pose is updated for each time step with the control inputs and the Lidar readings. As the robot moves, the particle cloud becomes denser and denser, reflecting the pose estimation converges and the confidence of the belief in robot pose increases. The robot stops once a statistics bond on the localization error is obtained and the output from AMCL is the fine-tuned robot pose.

4. Experimental results and discussion

4.1. Experimental setup

A mobile platform was used to test the proposed robot initialization system. The robot is a four-wheel differential drive, skid-steering robot with zero-degree turning radius, driven by 4 × 200 Watt brushless servo motors. The onboard processing is achieved by Nvidia Jetson AGX Xavier and a mounted industrial PC which is equipped with an NVIDIA GeForce RTX 2080 graphics card. The robot is equipped with a variety of perception sensors: a 3D lidar (RoboSense RS-LiDAR-16), an RGBD camera (Intel D455 RealSense), IMU, and wheel encoders.

The proposed robot initialization system was first validated in hallways at the university building at Nanyang Technological University (NTU), and then tested in residential buildings under construction. Two construction blocks with different floor plans at the site were selected for the experiments. All testing environments exhibit the characteristics of highly symmetric settings with long corridors, similar plain walls and pillars. The robot was initially deployed at 17 randomly selected locations in the NTU environment and 9 locations for each residential block. The scenes of testing environments and the robot starting positions are shown in Fig. 8. The dots and arrows represent the robot's starting positions and the heading directions, respectively. The robot was deployed at the corridors in the experiments since the robot was usually transported to the lift lobby or charging stations at the corridor to start the work. It navigated inside specific units only when necessary to ensure an effective navigation system. Moreover, the long corridor environments were more critical regions to be tested since the environment was highly repetitive with fewer features.

The hallway environment at NTU is 189.13 m long with two main narrow corridors that are 2 m wide and a few paths connecting the two long corridors. There are 38 doors and 5 lifts and each component is considered a distinctive feature. The two residential blocks have the

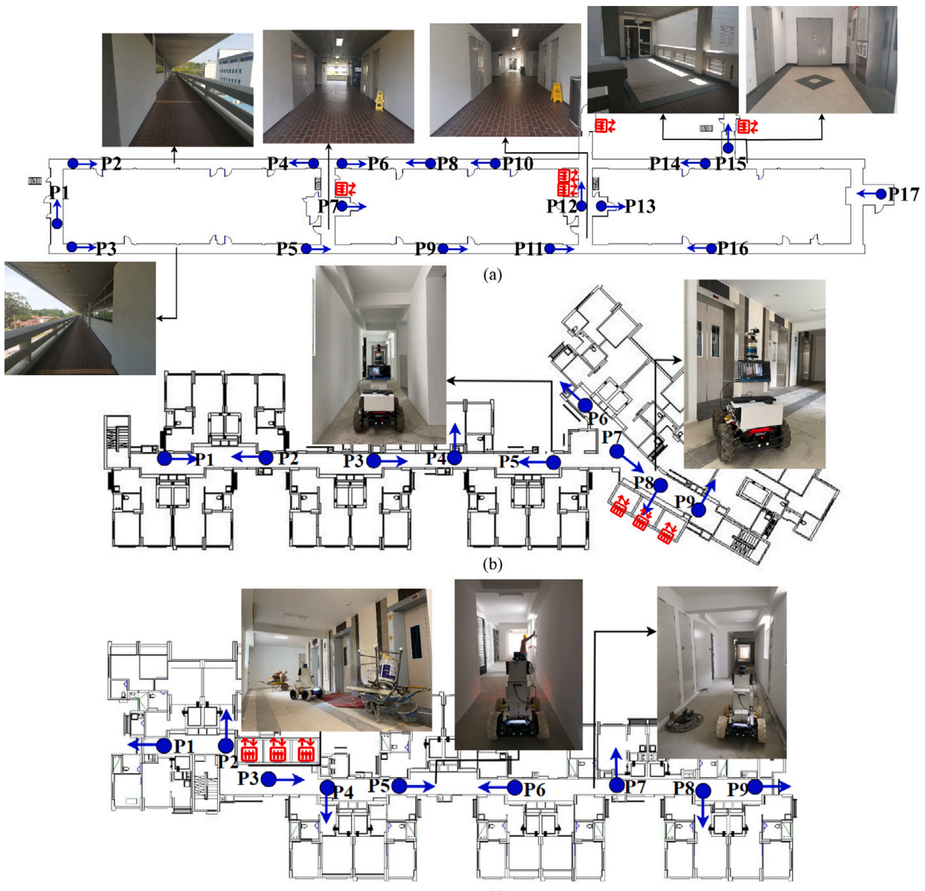


Fig. 8. Testing environments (a) NTU (b) Residential block A (c) Residential block B.

same appearance of architectural objects, which are lifts, staircases and main gates of the units. There are 13 main gates, 3 lifts and 2 staircases on one floor for each block. However, the layouts for the two blocks are different, making the geometric patterns among the aforementioned objects different. The lengths of the corridors for the two blocks are 61.3 m and 62.8 m respectively. Block A was in the final inspection stage while Block B involved more construction work, leaving more construction materials and moving workers in the corridor. The real industrial BIM models were provided by the project's main contractor and the models were used to retrieve the geometric and semantic information of the architectural components.

4.2. Experimental results

The robot started from the deployed position, explored the environment and matched detected features to the corresponding BIM features using the proposed online decision making strategy until the unique match was determined. Then the pose of the robot was estimated and rectified using (10), (13), and (17), (18). Among the 35 starting locations in the three environments, the correct matched feature sequences and the estimations of robot pose were found for all the trials. The number of detected features and moving distance for the coarse initialization, localization errors, the total moving distance after applying the PF-based fine tuning and the corresponding localization errors for different environments are presented in Fig. 9 and the mean values are summarized in Table 2.

In the NTU environment, the robot traveled for 44.88 m on average to complete the coarse initialization. The position was found after 3 or 4 features were detected in general and more features may be needed if a similar pattern of features appeared at multiple locations. It was shown that the number of feasible matches reduced as more features were detected since false matching candidates were eliminated. Taking starting position P3 as an example, Fig. 10 shows the process of filtering out wrong matching candidates. The four sub Figs. 10(a) (b) (c) (d) correspond to the possible moving paths for 3, 4, 5, 6 features respectively. When 3 features were detected, 26 feasible matches were generated. The 8 matches with the least matching cost are shown with arrows representing the robot path. As the robot continued to explore and the fourth feature was detected, the number of possible matches was reduced to less than 10. The four matches with the least matching cost are shown. There were only two possibilities left after the fifth feature was detected, and the unique matching was found with 6 features detected. Similar to NTU tests, the robot location and route can be predicted after detecting 3 to 7 visual features for residential construction environments. However, the moving distance for coarse localization and fine localization were around 15 m and 22 m respectively, which were less than half of those for NTU environments as the distance between main gates at residential blocks was around 5 m while it was 10 to 20 m between doors at NTU corridors.

After the coarse initialization, the BIM-based constraint analysis and PF-based fine tuning were applied to rectify the coarsely estimated robot pose. The robot localization result using the multi-level positioning rectification is illustrated in Fig. 11. As present in Fig. 9(c)–(f), the multi-level positioning rectification significantly improved the robot localization accuracy. In the NTU environment, the average translational and rotational errors for the coarse initialization were 1.32 m and 22.63 degrees respectively. The average errors were reduced to 0.61 m and 8.91 degrees after the rectification with the applied BIM geometric constraints and the accuracy was improved to 0.21 m and 1.47 degrees with the PF-based fine tuning. In residential construction environments,

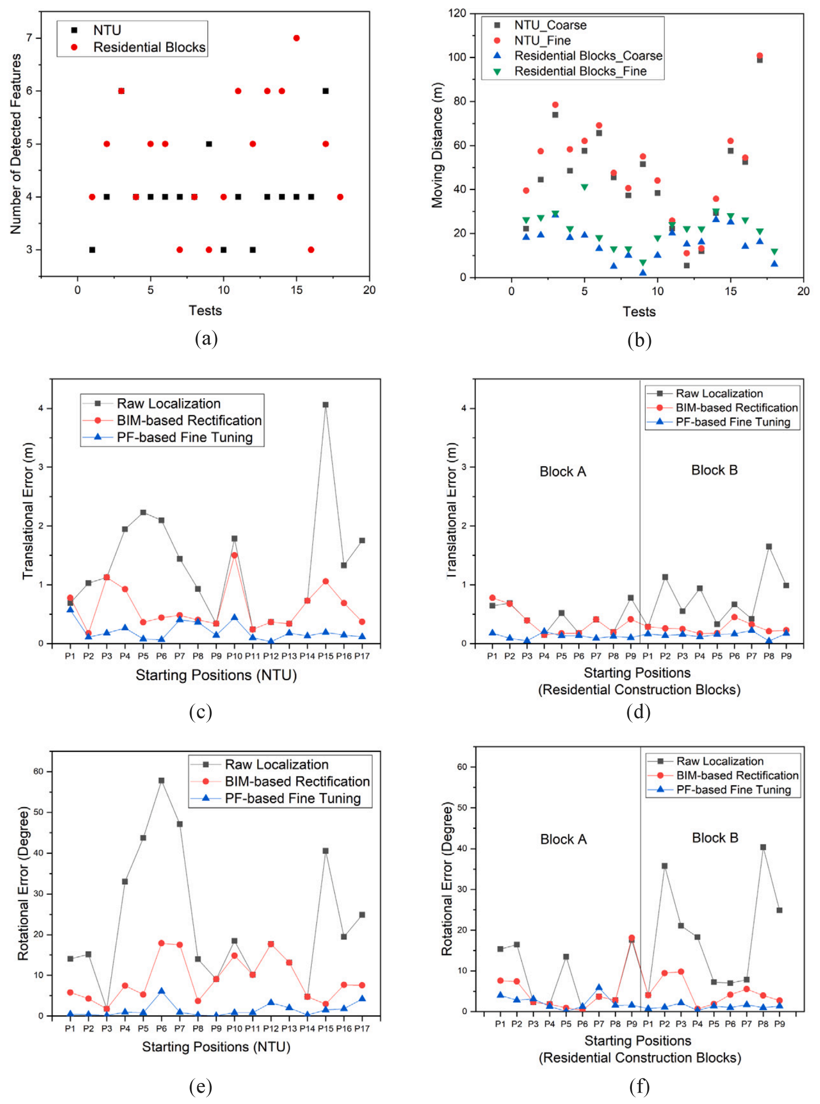


Fig. 9. Experimental results (a) Number of detected features for coarse initialization (b) Moving distances for coarse and fine initialization (c) Translational localization error in the NTU environment (d) Translational localization error in residential construction environments (e) Rotational localization error in the NTU environment (f) Rotational localization error in residential construction environments.

Table 2
Robot localization statistics for different environments.

Block	Average No. of detected features	Average moving distance (m)	Average BIM-based rectification error		Average total moving distance (m)	Average PF-based fine tuning error	
			Translational (m)	Rotational (°)		Translational (m)	Rotational (°)
NTU	4	44.88	0.61	8.91	50.32	0.21	1.47
BlockA	4	14.81	0.37	5.00	21.99	0.12	2.39
BlockB	5	16.61	0.26	4.69	22.76	0.15	1.18

the mean localization errors were reduced from 0.61 m and 13.35 degrees to around 0.3 m and 5 degrees with BIM-based rectification and 0.15 m and 2 degrees with PF-based fine tuning. Compared with the NTU environment, the positioning errors for residential blocks were lower in general, especially for the localization with BIM-based rectification as the corridors and lobby area at residential blocks were more narrow, providing more geometric constraints.

The feasibility of the proposed system was further validated by performing robot navigation that took the fine tuned initialization result as the initial pose. Navigation tests were conducted for the 9 trails at Block B and selected points at the NTU environment. The robot was able to navigate to the target point smoothly with the given initialization.

The testing results at different environments validated the generality of the robot initialization method in highly symmetric construction environments, including both industrial buildings like the university building, and residential buildings. Moreover, the BIM information retrieval application, the object detector and the robot motion planning algorithm were environment-independent. The proposed system was applied for the two residential blocks of different layouts and construction work without any additional pre-scanning. The experiments illustrated that the proposed system can be easily applied across different blocks and is useful for the robot platforms that are designed

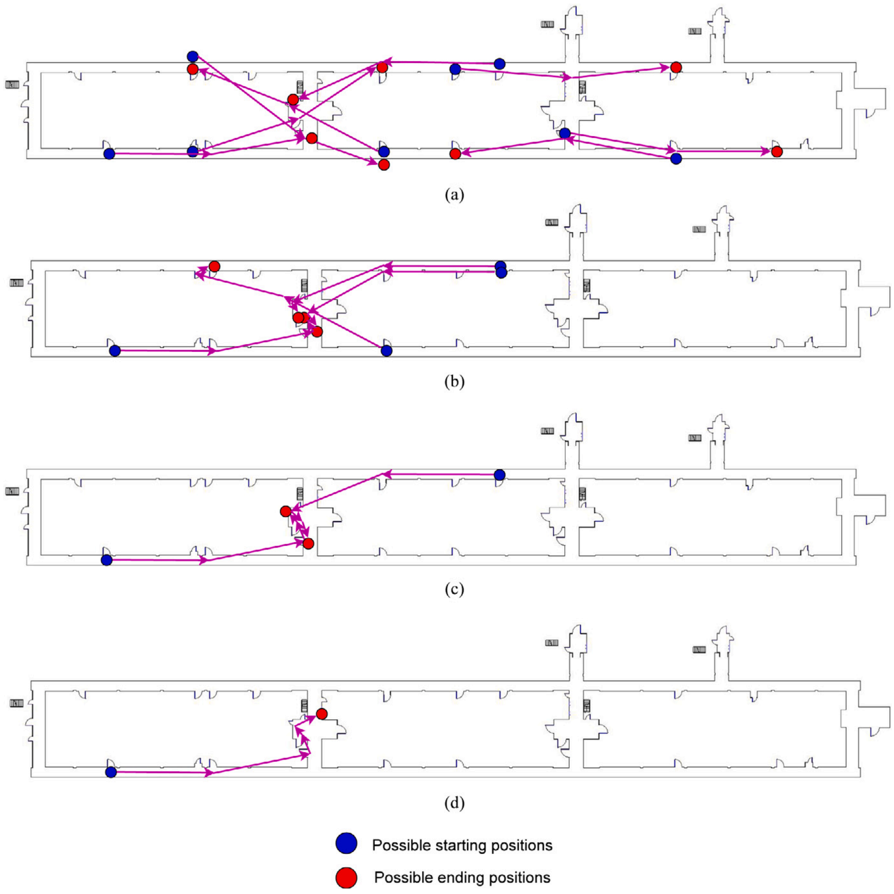


Fig. 10. Possible robot positions and paths with different numbers of features detected (a) 3 features (b) 4 features (c) 5 features (d) 6 features.

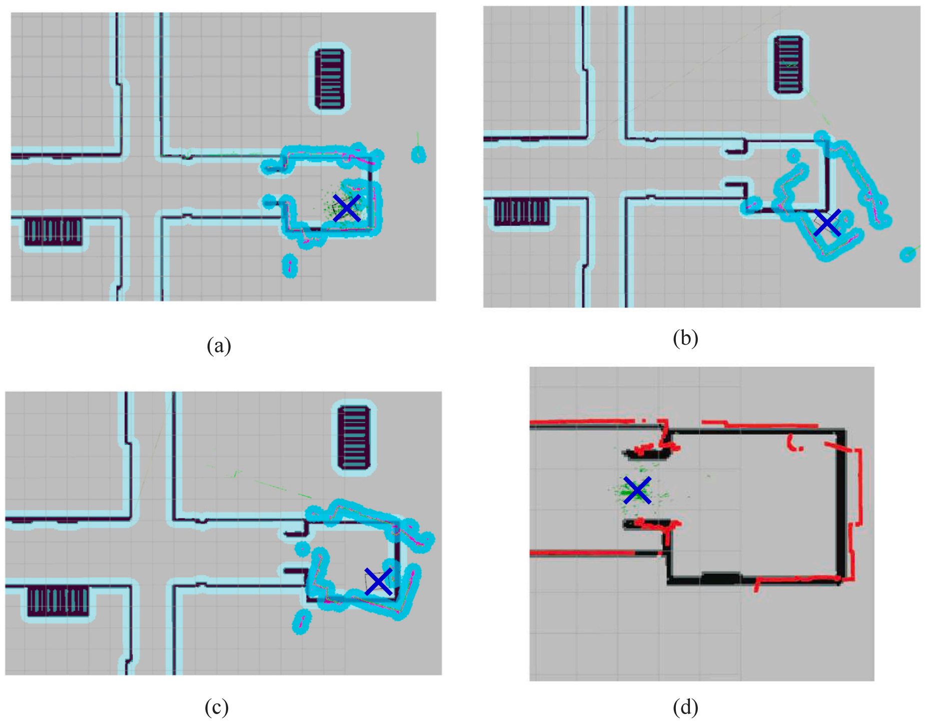


Fig. 11. Robot localization result (a) Ground truth (b) Coarse localization (c) BIM-based rectification (d) PF-based fine tuning.

Table 3
Robot localization statistics for different methods.

Methods	Success rate (%)	Average moving distance (m)	Average translational error (m)	Average rotational error (Degree)
BIM-based rectification	100	44.88	0.61	8.91
BIM-based + PF-based fine tuning	100	50.32	0.21	1.47
ArUco marker	94	51.21	0.17	2.75
AMCL	35	12.00	1.04	1.43

to work at multiple blocks at the same construction site. It provided the ease of robot initialization without any marker and infrastructure deployment, or the construction of the unique image or signal strength database.

4.3. Experimental evaluations

The proposed method was compared with two commonly used robot global localization methods to investigate its advantages and limitations. The marker-based method using ArUco tag [16] and the laser-based method, AMCL [35,38] global localization mode were implemented to localize the robot for the NTU environment from the same 17 starting positions using the same wall following based motion planning algorithm.

To implement the marker-based method, two ArUco tags were deployed for each section and the location was the exact middle of each corridor segment. Therefore, there were six markers in total, evenly distributed throughout the entire working environment. The 20 cm $\times$20 cm markers were printed on paper and pasted on the walls. The RGBD camera was used to detect the marker using OpenCV library and then estimate the robot location by decoding the unique marker.

The AMCL global localization mode was tested for the robot initialization by generating 5000 randomly distributed particles on the map to present the possible robot poses. As the robot moved, the probabilistic distribution of the robot pose was updated with the motion model and observation model. The particles were grouped into multiple particle clouds and each cloud represented one possible located region. The localization confidence level increased as the moving distance increased and the localization was considered completed when the covariance was lower than the predefined threshold. The selected threshold was the same as that for PF-based fine tuning in the proposed BIM-based initialization method.

Various criteria were applied to evaluate the initialization performance, including the success rate, average moving distance, and average localization error that reflected the robustness, efficiency, and accuracy respectively [39]. The recorded moving distance was the total distance the robot traveled to localize itself. The same criteria for successful localization was used for all the methods. The localization was considered as failed if the moving distance was larger than 150 m but the position could not be found yet, or the localization error was larger than 5 m. The initialization results for all the 17 starting positions using the proposed BIM-based initialization method with different levels of rectification, and the two comparable methods are present in Fig. 12. The initialization statistics is concluded in Table 3. Only the successful trials were used to calculate the average moving distance and localization error.

It had been seen that the proposed initialization method yielded the highest success rate of 100% while the ArUco marker method and AMCL method achieved the rate of 94% and 35% respectively. There was localization failure for one point with the ArUco tag method as the robot detected the marker from a far distance of over 10 m. There were large errors in the detection of corner locations of the small marker. Therefore, it significantly degraded the accuracy of robot localization. Regarding the AMCL method, the success rate was low since the global robot localization mode using particle filter without any prior knowledge of the environment performed poorly for repetitive environments

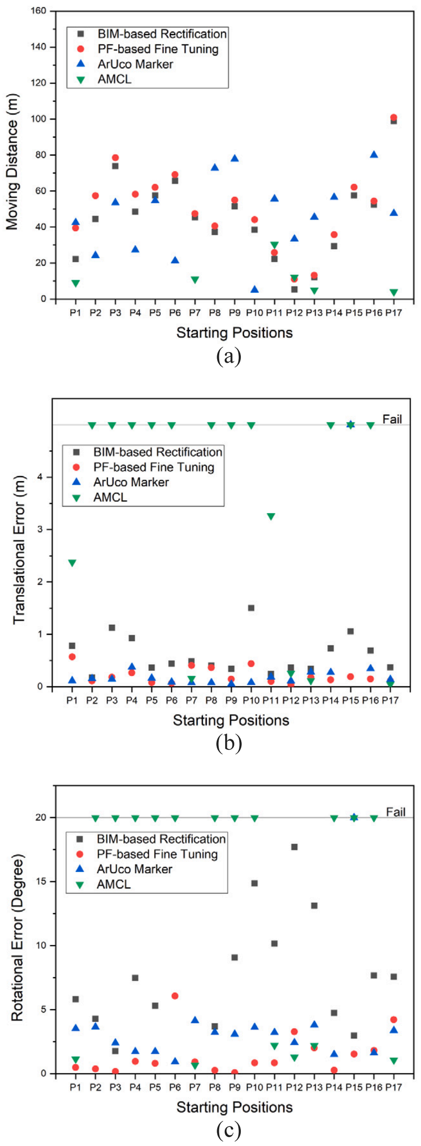
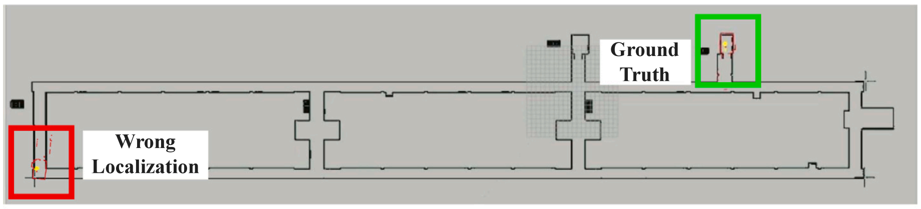


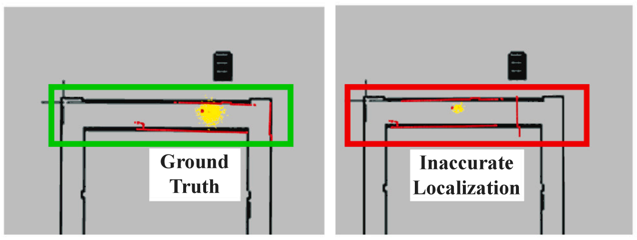
Fig. 12. Experimental evaluations with comparable methods (a) Moving distance (b) Translational localization error (c) Rotational localization error.

containing similar structures. For the experiments at NTU, it failed for most of the starting positions, especially for the ones starting in the long corridor where no distinctive geometric features can be observed. As similar geometric features appeared at various locations, although the particles often converged to a single pose hypothesis quickly, AMCL was vulnerable to wrong localization to other positions exhibiting similar characteristics [37], as shown in Fig. 13(a).

In terms of efficiency, localization using AMCL was the fastest with the shortest moving distance as it implemented a probabilistic based algorithm that retrieved information from each time step and updated



(a)



(b)

Fig. 13. Localization problems with AMCL method (a) Wrong localization (b) Inaccurate localization.

the position estimation continuously. It was capable of converging quickly. Since the proposed method and marker-based method both required the detection of discrete visual objects, the moving distance was larger and the moving distance also depended on how far every two features were separated. It was noticed that the average moving distances of the proposed method were slightly lower than the marker-based method. The robot missed the first ArUco marker for two trials due to the poor viewpoints and change of illumination as the robot moved. Therefore, it took a long distance to find the second marker if the first one encountered was not detected. The proposed method that detected common architectural features outperformed since it collected the information of all detected objects and utilized the information of relative positions. Although some objects along the way were not detected, it could still rely on the other detected objects to localize, rather than moving for a long time to find the unique marker. Moreover, both the marker-based and laser-based methods were passive methods where robot motion was not aimed for the localization task. The proposed BIM-based method utilized the BIM geometric and semantic information to guide the robot motion and performed active exploration and thus, managed to expedite the localization process and eliminate the wrong position candidates efficiently.

Regarding the localization error, it has been showed that PF-based fine tuning can greatly reduce the localization error compared to the BIM-based rectification method. The proposed method with fine tuning had comparable accuracy with the best performed methods in terms of both positioning and heading, giving 21 cm and 1.47 degrees error respectively. It was also noticed that the variation in positioning accuracy was large for AMCL, with some trials possessing a sub-cm error while others having 2 to 4 meters error. The reason was that although the robot could correctly find the region it fell in, there could be large offsets between the ground truth and estimated location in the corridor areas lacking sufficient features to rectify and align the estimation, as shown in Fig. 13(b). The boundness of error was not guaranteed for global localization using AMCL and it was influenced by the initial particle distribution [54]. In contrast, the BIM-based initialization with PF-based fine tuning applied the same AMCL algorithm for the rectification of localization but obtained more accurate positioning results since the coarse estimation of the pose was given by visual feature matching, rather than a random distribution.

In conclusion, the proposed method using BIM-based rectification and PF-based fine tuning was the most robust among these methods in the sense of achieving a higher success rate and was applicable for all the starting points while the other two baseline methods failed for certain scenarios. The proposed method with PF-based fine tuning possessed comparable localization accuracy with the most accurate positioning and heading results obtained from baseline methods. A poor success rate and a large average positioning error had been seen with AMCL global localization since wrong localization and large offsets presented in repetitive environments. The marker-based method provided acceptable localization accuracy but required additional effort on marker deployment and was prone to error with viewpoint and environment changes. Compared with the baseline methods, the proposed BIM-based method was infrastructure-free and robust, providing accurate localization results.

The proposed method is currently limited to 2D robot initialization using wheeled robots. The 3D information can be obtained if legged robots are used where the story information can be obtained through recorded stair climbing starting from ground floor. Otherwise, commercial communication devices can be mounted in the elevator to communicate with the robot to move it to any story. Precise barometers can also be installed on the robot for measurement of elevation. In addition, visual occlusions due to dynamic obstacles in the construction environments can result in missed object detection and prolong the traversing time for initialization. The proposed method is limited to a single mobile robot system and the supplementary sensor data from multi-robot coordination, e.g. a mobile robot-drone collaboration to overcome the occlusion can be integrated in the future for faster initialization.

5. Conclusions and future work

This paper presents a BIM-based mobile robot initialization system using object detection for construction automation. This method associates the acquired online sensory information of object detection with the known BIM information to realize the automatic robot initialization. The proposed system is composed of the offline BIM information retrieval, training of the CNN-based object detector, online feature detection and matching, BIM-based active exploration, robot decision making strategies, as well as the multi-level positioning rectification. It leverages BIM in various aspects to facilitate the robot initialization, expedite the initialization process, and improve the localization accuracy. Different from the traditional robot positioning and initialization methods, the proposed method is infrastructure-free and environment-independent. It eliminates the effort for device deployment and environment configuration for robot initialization in construction automation. The proposed work contributes to achieve the fully automated robot navigation for construction automation by introducing an automatic initialization approach.

The proposed autonomous robot initialization system was validated at a university building and two residential buildings that were under construction without infrastructure deployment and pre-scanning of the environments. The same BIM information retrieval application, object detector, and robot motion planning algorithms were applied for the two construction blocks with different scenes and structures, illustrating that the proposed system is environment-independent. The comparison with two robot global localization methods, including a marker-based method and a laser-based localization software, AMCL was conducted. The proposed method achieved a higher success rate on robot initialization for random starting points in highly repetitive environments. The proposed method using BIM-based rectification and PF-based fine tuning possessed comparable localization accuracy with the most accurate positioning and heading results obtained from baseline methods.

Future work can be performed to integrate the detection of low-level visual features that require no prior training, such as inner and outer wall corners with the existing object detection to achieve a faster coarse initialization. In addition, the proposed system in this paper can be extended to multi-robot initialization in construction automation. As a group of robots deployed at different locations in the same environment, they observe different local visual features. Information about the surrounding environment for each robot can be passed to other robots in its neighborhood. Equipped with the information-sharing scheme, data association between BIM information and observed online sensory information can be more efficient by integrating the information and synchronizing the robots' beliefs on their locations.

Data availability

The authors do not have permission to share data.

Acknowledgments

This work is supported by the Agency For Science, Technology and Research (A*STAR), Singapore, under the National Robotics Program (NRP)-Robotics Domain Specific (RDS: Ref. [1922200001]). We would like to acknowledge Teambuild Construction Group, Singapore, and Building and Construction Authority (BCA), Singapore for providing support and research resources.

References

[1] R. Kangari, D.W. Halpin, Potential robotics utilization in construction, J. Constr. Eng. Manage. 115 (1) (1989) 126–143, http://dx.doi.org/10.1061/(ASCE)0733-9364(1989)115:1(126).

[2] T. Bock, T. Linner, Construction Robots: Volume 3: Elementary Technologies and Single-Task Construction Robots, Cambridge University Press, 2016, ISBN: 1316785173.

[3] Z. Wang, H. Li, X. Zhang, Construction waste recycling robot for nails and screws: Computer vision technology and neural network approach, Autom. Constr. 97 (2019) 220–228, http://dx.doi.org/10.1016/j.autcon.2018.11.009.

[4] M. Ilyas, H.Y. Khaw, N.M. Selvaraj, Y. Jin, X. Zhao, C.C. Cheah, Robot-assisted object detection for construction automation: Data and information-driven approach, IEEE/ASME Trans. Mechatronics 26 (6) (2021) 2845–2856, http://dx.doi.org/10.1109/TMECH.2021.3100306.

[5] R.-J. Yan, E. Kayacan, I.-M. Chen, L.K. Tiong, J. Wu, QuicaBot: Quality inspection and assessment robot, IEEE Trans. Autom. Sci. Eng. 16 (2) (2019) 506–517, http://dx.doi.org/10.1109/TASE.2018.2829927.

[6] M.M. Atia, A. Noureldin, M.J. Korenberg, Dynamic online-calibrated radio maps for indoor positioning in wireless local area networks, IEEE Trans. Mob. Comput. 12 (9) (2013) 1774–1787, http://dx.doi.org/10.1109/TMC.2012.143.

[7] Y. Zhuang, J. Yang, Y. Li, L. Qi, N. El-Sheimy, Smartphone-based indoor localization with bluetooth low energy beacons, Sensors 16 (5) (2016) pp. 596, http://dx.doi.org/10.3390/s16050596.

[8] H. Kawaji, K. Hatada, T. Yamasaki, K. Aizawa, Image-based indoor positioning system: Fast image matching using omnidirectional panoramic images, in: Proceedings of the 1st ACM International Workshop on Multimodal Pervasive Video Analysis, Association for Computing Machinery, 2010, pp. 1–4, http://dx.doi.org/10.1145/1878039.1878041.

[9] M. Bosse, R. Zlot, Keypoint design and evaluation for place recognition in 2D lidar maps, Robot. Auton. Syst. 57 (12) (2009) 1211–1224, http://dx.doi.org/10.1016/j.robot.2009.07.009.

[10] C. Yang, H.-r. Shao, WiFi-based indoor positioning, IEEE Commun. Mag. 53 (3) (2015) 150–157, http://dx.doi.org/10.1109/MCOM.2015.7060497.

[11] M. Nowicki, J. Wietrzykowski, Low-effort place recognition with WiFi fingerprints using deep learning, in: R. Szewczyk, C. Zieliński, M. Kaliczyńska (Eds.), Automation 2017, Springer International Publishing, pp. 575–584, http://dx.doi.org/10.1007/978-3-319-54042-9_57.

[12] S. Woo, S. Jeong, E. Mok, L. Xia, C. Choi, M. Pyeon, J. Heo, Application of WiFi-based indoor positioning system for labor tracking at construction sites: A case study in guangzhou MTR, Autom. Constr. 20 (1) (2011) 3–13, http://dx.doi.org/10.1016/j.autcon.2010.07.009.

[13] J. González, J. Blanco, C. Galindo, A.O. de Galisteo, J. Fernández-Madrigal, F. Moreno, J. Martínez, Mobile robot localization based on ultra-wide-band ranging: A particle filter approach, Robot. Auton. Syst. 57 (5) (2009) 496–507, http://dx.doi.org/10.1016/j.robot.2008.10.022.

[14] S. Han, H. Lim, J. Lee, An efficient localization scheme for a differential-driving mobile robot based on RFID system, IEEE Trans. Ind. Electron. 54 (6) (2007) 3362–3369, http://dx.doi.org/10.1109/TIE.2007.906134.

[15] Z. Li, J. Huang, Study on the use of Q-R codes as landmarks for indoor positioning: Preliminary results, in: 2018 IEEE/ION Position, Location and Navigation Symposium, 2018, pp. 1270–1276, http://dx.doi.org/10.1109/PLANS.2018.8373516.

[16] S. Garrido-Jurado, R. Muñoz-Salinas, F.J. Madrid-Cuevas, M.J. Marín-Jiménez, Automatic generation and detection of highly reliable fiducial markers under occlusion, Pattern Recognit. 47 (6) (2014) 2280–2292, http://dx.doi.org/10.1016/j.patcog.2014.01.005.

[17] S.-J. Lee, G. Tewolde, J. Lim, J. Kwon, QR-code based localization for indoor mobile robot with validation using a 3D optical tracking instrument, in: 2015 IEEE International Conference on Advanced Intelligent Mechatronics, 2015, pp. 965–970, http://dx.doi.org/10.1109/AIM.2015.7222664.

[18] G. Goronzy, M. Pelka, H. Hellbrück, QRPos: Indoor positioning system for self-balancing robots based on QR codes, in: 2016 International Conference on Indoor Positioning and Indoor Navigation, 2016, pp. 1–8, http://dx.doi.org/10.1109/IPIN.2016.7743616.

[19] A.J. Davison, I.D. Reid, N.D. Molton, O. Stasse, MonoSLAM: Real-time single camera SLAM, IEEE Trans. Pattern Anal. Mach. Intell. 29 (6) (2007) 1052–1067, http://dx.doi.org/10.1109/TPAMI.2007.1049.

[20] G. Klein, D. Murray, Parallel tracking and mapping for small AR workspaces, in: 2007 6th IEEE and ACM International Symposium on Mixed and Augmented Reality, 2007, pp. 225–234, http://dx.doi.org/10.1109/ISMAR.2007.4538852.

[21] R. Mur-Artal, J.M.M. Montiel, J.D. Tardós, ORB-SLAM: A versatile and accurate monocular SLAM system, IEEE Trans. Robot. 31 (5) (2015) 1147–1163, http://dx.doi.org/10.1109/TRO.2015.2463671.

[22] R. Mur-Artal, J.D. Tardós, ORB-SLAM2: An open-source SLAM system for monocular, stereo, and RGB-D cameras, IEEE Trans. Robot. 33 (5) (2017) 1255–1262, http://dx.doi.org/10.1109/TRO.2017.2705103.

[23] S. Sumikura, M. Shibuya, K. Sakurada, OpenVSLAM: A versatile visual SLAM framework, in: Proceedings of the 27th ACM International Conference on Multimedia, Association for Computing Machinery, 2019, pp. 2292–2295, http://dx.doi.org/10.1145/3343031.3350539.

[24] P.-Y. Tseng, J.J. Lin, Y.-C. Chan, A.Y. Chen, Real-time indoor localization with visual SLAM for in-building emergency response, Autom. Constr. 140 (2022) pp. 104319, http://dx.doi.org/10.1016/j.autcon.2022.104319.

[25] Y. Bai, W. Jia, H. Zhang, Z.-H. Mao, M. Sun, Landmark-based indoor positioning for visually impaired individuals, in: 2014 12th International Conference on Signal Processing, ICSP, 2014, pp. 668–671, http://dx.doi.org/10.1109/ICOSP.2014.7015087.

[26] K. Wan, L. Ma, X. Tan, An improvement algorithm on RANSAC for image-based indoor localization, in: 2016 International Wireless Communications and Mobile Computing Conference, IWCMC, IEEE, pp. 842–845, http://dx.doi.org/10.1109/IWCMC.2016.7577167.

[27] W. Zhang, G. Liu, G. Tian, A coarse to fine indoor visual localization method using environmental semantic information, IEEE Access 7 (2019) 21963–21970, http://dx.doi.org/10.1109/ACCESS.2019.2899049.

[28] H. Bay, A. Ess, T. Tuytelaars, L. Van Gool, Speeded-up robust features (SURF), Comput. Vis. Image Underst. 110 (3) (2008) 346–359, http://dx.doi.org/10.1016/j.cviu.2007.09.014.

[29] S. Xu, W. Chou, H. Dong, A robust indoor localization system integrating visual localization aided by CNN-based image retrieval with Monte Carlo localization, Sensors 19 (2) (2019) pp. 249, http://dx.doi.org/10.3390/s19020249.

[30] G. Grisetti, C. Stachniss, W. Burgard, Improved techniques for grid mapping with Rao-blackwellized particle filters, IEEE Trans. Robot. 23 (1) (2007) 34–46, http://dx.doi.org/10.1109/TRO.2006.889486.

[31] S. Kohlbrecher, O. von Stryk, J. Meyer, U. Klingauf, A flexible and scalable SLAM system with full 3D motion estimation, in: 2011 IEEE International Symposium on Safety, Security, and Rescue Robotics, 2011, pp. 155–160, http://dx.doi.org/10.1109/SSRR.2011.6106777.

[32] W. Hess, D. Kohler, H. Rapp, D. Andor, Real-time loop closure in 2D LIDAR SLAM, in: 2016 IEEE International Conference on Robotics and Automation, ICRA, 2016, pp. 1271–1278, http://dx.doi.org/10.1109/ICRA.2016.7487258.

[33] E. Olson, M3RSM: Many-to-many multi-resolution scan matching, in: 2015 IEEE International Conference on Robotics and Automation, ICRA, 2015, pp. 5815–5821, http://dx.doi.org/10.1109/ICRA.2015.7140013.

[34] D. Fox, W. Burgard, S. Thrun, Markov localization for mobile robots in dynamic environments, J. Artificial Intelligence Res. 11 (1999) 391–427, http://dx.doi.org/10.1613/jair.616.

[35] F. Dellaert, D. Fox, W. Burgard, S. Thrun, Monte Carlo localization for mobile robots, in: Proceedings 1999 IEEE International Conference on Robotics and Automation (Cat. No.99CH36288C), Vol. 2, 1999, pp. 1322–1328 vol.2, http://dx.doi.org/10.1109/ROBOT.1999.772544.

[36] D. Fox, Adapting the sample size in particle filters through KLD-sampling, Int. J. Robot. Res. 22 (12) (2003) 985–1003, http://dx.doi.org/10.1177/0278364903022012001.

[37] Z. Liu, Z. Shi, M. Zhao, W. Xu, Mobile robots global localization using adaptive dynamic clustered particle filters, in: 2007 IEEE/RSJ International Conference on Intelligent Robots and Systems, 2007, pp. 1059–1064, http://dx.doi.org/10.1109/IROS.2007.4399050.

[38] B.P. Gerkey, AMCL, 2020, URL http://wiki.ros.org/amcl.

[39] Z. Wu, J. Zhang, Y. Yue, M. Wen, Z. Jiang, H. Zhang, D. Wang, Infrastructure-free global localization in repetitive environments: An overview, in: IECON 2020 the 46th Annual Conference of the IEEE Industrial Electronics Society, 2020, pp. 626–631, http://dx.doi.org/10.1109/IECON43393.2020.9255046.

[40] M. Xu, S. Wei, S. Zlatanova, R. Zhang, BIM-Based indoor path planning considering obstacles, ISPRS Ann. Photogramm. Rem. Sensing Spatial Inf. Sci. IV-2/W4 (2017) 417–423, http://dx.doi.org/10.5194/isprs-annals-IV-2-W4-417-2017.

[41] D. Acharya, M. Ramezani, K. Khoshelham, S. Winter, BIM-tracker: A model-based visual tracking approach for indoor localisation using a 3D building model, ISPRS J. Photogramm. Remote Sens. 150 (2019) 157–171, http://dx.doi.org/10.1016/j.isprsjprs.2019.02.014.

[42] A. Hamieh, A. Ben Makhlouf, B. Louhichi, D. Deneux, A BIM-based method to plan indoor paths, Autom. Constr. 113 (2020) pp. 103120, http://dx.doi.org/10.1016/j.autcon.2020.103120.

[43] J. Park, J. Chen, Y.K. Cho, Self-corrective knowledge-based hybrid tracking system using BIM and multimodal sensors, Adv. Eng. Inf. 32 (2017) 126–138, http://dx.doi.org/10.1016/j.aei.2017.02.001.

[44] N. Li, B. Becerik-Gerber, B. Krishnamachari, L. Soibelman, A BIM centered indoor localization algorithm to support building fire emergency response operations, Autom. Constr. 42 (2014) 78–89, http://dx.doi.org/10.1016/j.autcon.2014.02.019.

[45] I. Ha, H. Kim, S. Park, H. Kim, Image retrieval using BIM and features from pretrained VGG network for indoor localization, Build. Environ. 140 (2018) 23–31, http://dx.doi.org/10.1016/j.buildenv.2018.05.026.

[46] D. Acharya, K. Khoshelham, S. Winter, BIM-PoseNet: Indoor camera localisation using a 3D indoor model and deep learning from synthetic images, ISPRS J. Photogramm. Remote Sens. 150 (2019) 245–258, http://dx.doi.org/10.1016/j.isprsjprs.2019.02.020.

[47] M.S. Moura, C. Rizzo, D. Serrano, BIM-based localization and mapping for mobile robots in construction, in: 2021 IEEE International Conference on Autonomous Robot Systems and Competitions, ICARSC, 2021, pp. 12–18, http://dx.doi.org/10.1109/ICARSC52212.2021.9429779.

[48] J. Redmon, A. Farhadi, Yolov3: An incremental improvement, 2018, http://dx.doi.org/10.48550/ARXIV.1804.02767, ArXiv Preprint ArXiv:1804.02767.

[49] A. Bochkovskiy, C.-Y. Wang, H.-Y.M. Liao, Yolov4: Optimal speed and accuracy of object detection, 2020, http://dx.doi.org/10.48550/ARXIV.2004.10934, ArXiv Preprint ArXiv:2004.10934 (2020).

[50] J. Redmon, S. Divvala, R. Girshick, A. Farhadi, You only look once: Unified, real-time object detection, in: 2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR, 2016, pp. 779–788, http://dx.doi.org/10.1109/CVPR.2016.91.

[51] Y.-C. Chiu, C.-Y. Tsai, M.-D. Ruan, G.-Y. Shen, T.-T. Lee, Mobilenet-SSDv2: An improved object detection model for embedded systems, in: 2020 International Conference on System Science and Engineering, ICSSE, 2020, pp. 1–5, http://dx.doi.org/10.1109/ICSSE50014.2020.9219319.

[52] P. Virtanen, R. Gommers, T.E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright, S.J. van der Walt, M. Brett, J. Wilson, K.J. Millman, N. Mayorov, A.R.J. Nelson, E. Jones, R. Kern, E. Larson, C.J. Carey, İ. Polat, Y. Feng, E.W. Moore, J. VanderPlas, D. Laxalde, J. Perktold, R. Cimrman, I. Henriksen, E.A. Quintero, C.R. Harris, A.M. Archibald, A.H. Ribeiro, F. Pedregosa, P. van Mulbregt, SciPy 1.0 Contributors, SciPy 1.0: Fundamental algorithms for scientific computing in Python, Nature Methods 17 (2020) 261–272, http://dx.doi.org/10.1038/s41592-019-0686-2.

[53] V. Lumelsky, A. Stepanov, Dynamic path planning for a mobile automaton with limited information on the environment, IEEE Trans. Automat. Control 31 (11) (1986) 1058–1063, http://dx.doi.org/10.1109/TAC.1986.1104175.

[54] S. Thrun, Probabilistic robotics, Commun. ACM 45 (3) (2002) 52–57, http://dx.doi.org/10.1145/504729.504754.