A BIM-integrated Indoor Path Planning Framework for Unmanned Ground Robot

Zhengyi Chen[a], Keyu Chen[b], and Jack C. P. Cheng[a]

[a]Department of Civil and Environmental Engineering, The Hong Kong University of Science and Technology
[b]College of Civil Engineering and Architecture, Hainan University, Haikou, China
E-mail: zchenfq@connect.ust.hk, kchenal@connect.ust.hk, cejcheng@ust.hk

Abstract

Featuring manual-intensive labor as the primary source of productivity, the construction industry is plagued with low efficiencies and high safety risks. Increasing ground robots have the potential to address these shorting comings. However, the complicated indoor sites result in many barriers for adopting unmanned ground robots (UGR), especially in path planning. As the basis for the robot movement, reliable indoor path planning is often unavailable due to the lack of up-to-date maps and reliable path planning algorithms. This paper presents a BIM-integrated indoor path planning framework for UGRs. First, robot information is integrated into BIM as a knowledge base. Second, an IFC-based layered mapping is proposed to build a robot-centric prior map. After extracting necessary robot properties from the BIM base, an improved A* path planning algorithm is introduced to be adaptive to UGRs with different mobilities. Final simulation results have proven the value of this framework.

Keywords –

Building information modeling, ground robot mobility, layer mapping, Adaptive A* path planning

1 Introduction

The architecture, engineering, construction (AEC) industry is one of the main economics worldwide, but with various problems due to the limited automation, such as inefficiency and high safety risks. With the rapid development of computers and sensors, robotic system has become a promising method to automate the AEC industry. Among various robotics, unmanned ground robot (UGR) achieves the most attention in the indoor environment. It can provide not only reliable control in cluttered space but also high payload capacity to carry multiple sensors. Previous studies have proven UGR's advantages in AEC-related projects. For instance, [14] used a UGR for automatic data collection in hazardous areas, which reduced potential safety problems. Another UGR was proposed for removing materials to reduce labor costs [3].

Efficient path planning is crucial for UGR's navigation, aiming to find an optimal and robust path in different indoor environments. However, two main hindering factors still exist. First, buildings are becoming increasingly complex due to irregular geometries, diverse space functions, and dynamic changes [9], which poses a significant challenge for UGRs to acquire the knowledge of building data. Simultaneous localization and mapping (SLAM) is the classical mapping approach adopted by UGRs, but expensive sensors and manual inspections are required for constructing highly accurate maps [22], especially in complicated environments. Recently, reinforcement learning (RL) is proposed as a map-free method for UGR's navigation [7]. Although without the mapping requirement, RL requires costly real-world training, since models trained in the simulation are not guaranteed to be transferred for a real robot [22]. Second, UGR's mobility systems are not fully studied with previous path planning algorithms, while general ones (e.g. A*) may fail due to certain kinematic constraints. Generally, only one kind of UGR mobility system will be considered in each path planning algorithm, like the Ackerman system [11] or the skid-wheeling system [8]. A comprehensive framework that takes different UGR's mobilities into account is still in lack.

As a central database for the AEC industry, BIM contains not only geometric, semantic information of indoor elements, but also other project-related resources, like schedule, cost, equipment, etc. Thus, BIM could be extended to an N-dimensional platform for different purposes. In addition, BIM outperforms other computer-aided-design (CAD) tools in its object-oriented data format, namely "Industry Foundation Classes" (IFC), which advances the data exchange due to its high interoperability [21]. Due to these advantages, this study aims to propose a BIM-integrated framework to realize comprehensive indoor path planning for UGRs. The method initially integrates robot information (e.g. 3D

models, properties) into BIM as a knowledge base. Then, the IFC-based layered mapping is executed to construct a robot-centric prior map, which also establishes the relationship between BIM and the map of path planning. After extracting necessary robot properties from BIM, an adaptive A* path planning algorithm is introduced to improve the efficiency for UGRs with different mobility systems.

This paper is organized as follows: Section 2 reviews related studies about indoor environment mapping and path planning algorithms. Section 3 illustrates the proposed methods on the integration of BIM and robot information, the IFC-based layered mapping, and an adaptive A* path planning algorithm, followed by the validation design and results in Section 4. In closing, the conclusion and future work are presented in Section 5.

2 Literature Review

2.1 BIM-based indoor mapping

While not widely used in the robot domain, BIM has already been a popular resource for path planning mapping of other fields. Geometric information of BIM element is the basis for indoor mapping, which indicates its size and location. Follini, et al. [8] extracted the bounding box of physical objects from IFC files to get the lowest height, only those lower than height of moving object was considered as obstacles. For complex elements, Xu, et al. [26] proposed to use Boundary-representation (B-rep) to represent shapes, where a face is a bounded portion of a surface, and an edge is a bounded piece of curve. In addition, Brown, et al. [4] suggested that topological information should be extracted for indoor mapping, which could create an abstract and simplified network. Semantic information is what makes BIM different from other CAD software, generally including standard properties (e.g. name, construction phase and material) and other customized properties. For example, the construction data was parsed from IFC files by Follini, et al. [8], leading to a 4D dynamic BIM. Besides, Property Definition mechanism of BIM allows users to attach user-customized properties with Property Sets [17]. To express the status of BIM elements, Xu, et al. [26] added the property of public or private for *IfcSpace* and the property of locked or unlocked for *IfcDoor*. Li, et al. [15] linked indoor sensors into BIM to develop a real-time path planning platform for fire evacuation. Cheng, et al. [5] used CCTVs to measure the real-time congestion degree of a corridor to get a dynamic path for fire evacuation.

BIM offers not only valuable built-in information but also access to expand additional information. However, previous studies mainly focus on the BIM-based indoor mapping of facility management or safety planning, while a specific BIM environment for robot path planning is still lack.

2.2 Robot path planning

After constructing the indoor map, a path planning algorithm needs to be implemented for the optimal path. Firstly, the objectives of path planning should be determined, which are generally the components of the path's cost function. Boguslawski, et al. [2] optimized the path planning based on three criteria: hazard proximity, distance/ travel time, and route complexity. Similarly, Soltani, et al. [23] suggested a multi-objective framework for path planning at construction sites, which considered the transport cost, safety, and visibility to automate the routing analysis of people and vehicles during construction. In the context of robotic studies, mobility-related factors are drawn attention by researchers. Jun, et al. [12] and Jeong, et al. [11] considered the robot's stability during path planning on uneven ground surfaces. Zhou, et al. [27] used the control input of the robot and traveling time to construct a straight but efficient cost function.

The path planning algorithm is then adopted to find the optimal path based on above-mentioned criteria. Sampling-based methods are efficient in large environments, which generate random samples and connect to a search graph. Rapid-exploring Random Tree (RRT), RRT* and Probabilistic Road Map (PRM) are common sample-based examples [20]. Search-based methods, A* algorithm, developed by Hart, et al. [10], is also frequently used. Variants of A* algorithms have been developed for variable needs. Weighted A* was designed by Pohl [19] to adjust the weights of the heuristic function to reduce the planning time but easily getting into the local minimum. Aine, et al. [1] introduced MHA* to explore the optimum solution by using multiple heuristics simultaneously, while is time-consuming. Some studies also considered motion constraints. Lin, et al. [16] used straights and right/left turning as motion primitives of the RRT algorithm, while Dolgov, et al. [6] put forward Hybrid A* that considers the nonholonomic constraints of unmanned vehicles.

Turning back to the topic of UGR's indoor path planning, it could be found UGR's properties are usually ignored or partially considered in path planning. A UGR-centric algorithm deserves to be developed to fully cover the whole path planning process, including primitive generation, collision checking and cost setting.

3 Proposed Methods

This section describes the proposed methods. An

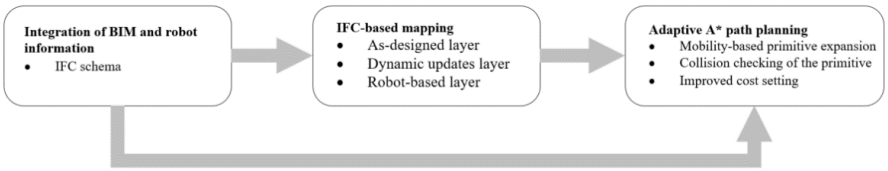


Figure 1. Workflow of the proposed methods

overview of the framework is illustrated in Figure 1.

3.1 Integration of BIM and robot information

As mentioned in section 2.1, BIM plays as an integrated platform to combine different object-related information. It can not only organize information more clearly but also provide interconnection and interoperability for decision makings. Referring to [24], the *IfcBuildingElementProxy* entity is a proxy definition with the same functionality as an *IfcBuildingElement* (i.e. walls, doors), so that it is herein adopted to represent robot models. Three IFC property sets are customized for the UGR by *IfcPropertySet*, namely Robot-Kinematic, Robot-Task and Robot-Dimensions. Further, specific properties are defined for each set by *IfcPropertySingleValue* (as shown in Figure 2). The final content of UGR's IFC entity is provided in Table 1.

```
#18777= IFCPROPERTYSINGLEVALUE('MaxSpeed (km/h)',$,IFCLENGTHMEASURE(5.),$);
#18778= IFCPROPERTYSINGLEVALUE('MaxTurningAngles (\X2\00B0\X0\)',$,IFCLENGTHMEASURE(40.),$);
#18779= IFCPROPERTYSINGLEVALUE('MaxBrakeTorch (N\X2\00B7\X0\m)',$,IFCLENGTHMEASURE(1500.),$);
#18780= IFCPROPERTYSINGLEVALUE('MaxTorch (N\X2\00B7\X0\m)',$,IFCLENGTHMEASURE(1000.),$);
#18781= IFCPROPERTYSINGLEVALUE('Mobility',$,IFCTEXT('Ackerman-Steering'),$);
#18782= IFCPROPERTYSINGLEVALUE('Weight (kg)',$,IFCLENGTHMEASURE(150.),$);
#18783= IFCPROPERTYSINGLEVALUE('SafetyBuffer (cm)',$,IFCLENGTHMEASURE(0.),$);
#18784= IFCPROPERTYSET('0jvwUs8gvDc9$NvHKPI Ey',#41,'Robot-Kinematic',$,(#18777,#18778,#18779,#18780,#18781,#18782,#18783));
```

Figure 2. Part of IFC definition for robot's dimensions

Table 1. IFC-based robot properties

Property Type	Property	Units
Robot-Kinematic	MaxSpeed	(km/h)
	MaxTurningAngles	°
	MaxBrakeTorch	N • m
	MaxTorch	N • m
	Mobility	Text
	Weight	Kg
	SafetyBuffer	cm
Robot-Task	StartingPoint	(m, m)
	StartOrientation	°
	EndingPoint	(m, m)
	StartingTime	yyyy/mm/dd/ HH/MM
Robot-Dimensions	Height	cm
	Length	cm
	Width	cm
	PivotToFront	cm
	PivotToFrontWheel	cm
	PivotToRealWheel	cm

Finally, *IfcFacetedBrep*, a manifold solid brep with the restriction that all faces are planar and bounded polygons, is used as the shape representation method to integrate external robot model resources (i.e. from SolidWorks or 3DMax) with BIM.

3.2 IFC-based mapping

The grid-based map is a well-known spatial model for path planning, which tessellates the indoor 3D space into a finite number of square cells. However, a single grid map is hard to clarify different information, where various data sources share a common map [18]. Thus, a multi-layered grid map is proposed to separate the processing of BIM data into semantical layers (Figure 3), which is suitable for the project's whole life cycle. Details of each layer and its corresponding BIM sources are as follows:

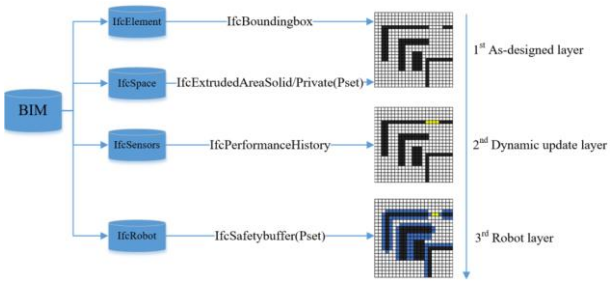


Figure 3. IFC-based layered mapping

1st layer: As-designed layer. The As-designed layer is a shared map for each robot originated from the BIM design plans, indicating physically passable areas. First, the current phase of the project should be determined to identify related indoor elements. This study selected the facility management phase for the demonstration, and floors, walls, doors and columns are included as the relevant building elements. However, our method is insensitive to other phases, like the construction phase which contains temporary facilities like formwork, scaffolding. Second, the *IfcExtrudedAreaSolid* of *IfcSpace* is extracted from the BIM models, which indicates the localization and the size of the space element. *IfcSpace's* extended property "public/private" is also required, where public space is constructed as the free space and private space is grouped to the obstacles. Finally, the *IfcBoundingBox* of *IfcWall*, *IfcDoor* and *IfcColumn* is drawn, composed of the world coordinates of each element's the lower-left corner and the upper

right corner. The areas covered by the *IfcBoundingBox* are considered as obstacles. Due to the trade-off between accuracy and computation load, the minimum width of *IfcBoundingBox* is taken as the unit grid size of the map, while the size of the map is decided by the boundary of *IfcSlab*. Hence, a preliminary 2D (*x*, *y*) grid cell map could be established, where an object's world coordinate (X, Y) should be connected to the cell coordinate (*x*, *y*) it arrives.

2nd layer: Dynamic updates layer. Path planning can also be affected by different dynamic factors, such as crowds, fire, locked doors. Dynamic updates refer to changes to the as-designed layer with real-time variations, mainly originated from sensors or construction schedules. Previous studies have demonstrated how to integrate real-time information with BIM [8,25], but such operations are out of the scope of this study. Hence, only the real-time status of door-related *IfcSensor* is considered in this article for the demonstration. In specific, the "locked" status of *IfcSensor* indicates cells covered by the door remain as obstacles, while cells covered by "unlocked" doors are turned to free space.

3rd layer: Robot-based layer. As the last step, this layer enables us to realize robot-centric mapping based on the robot data extracted from the IFC file. Each robot can obtain a unique map after customized refinements. Firstly, the robot's height will be compared with the bottom height of each building element, cells covered by elements higher than the robot will update their cost to 0. Besides, inflation sets cost value at the cells around obstacles according to the robot's SafetyBuffer.

3.3 Adaptive A* path planning

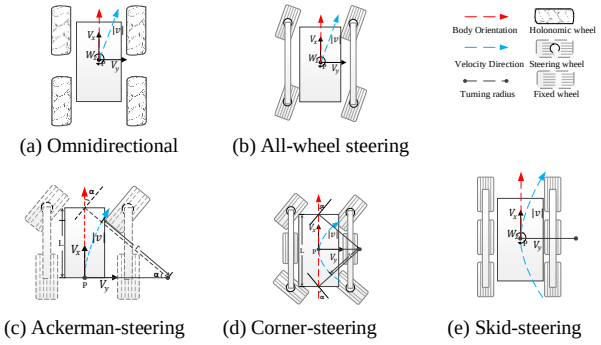
Our path planning module is a variant of the well-known A* algorithm that is adaptive to different kinds of UGR mobility systems. As shown in Alg.1, the searching loop is similar to the standard A* algorithm, where $\mathcal{O}$ and $\mathcal{C}$ refer to the open and closed set. Continuous motion primitives respecting the UGR's mobility are used as graph edges during ***PrimitiveExpansion()***, resulting in a structure *node* recording the pose information (Section 3.3.1). Then ***CollisionChecking()*** checks the feasibility of generated nodes (Section 3.3.2), only those with the smallest total cost (***PrimitiveCost, HeuristicCost***) will be saved (Section 3.3.3). This process continues until any primitive reach goal.

Algorithm 1: Adaptive A*

Initialize();
while $\neg\mathcal{O}$.**empty**() **do**
  $n_{current} \leftarrow \mathcal{O}$.**PopMin**(), $\mathcal{C}$.**insert**($n_{current}$);
  if **ReachGoal**($n_{current}$) **then**
    return **RetrievePath**();
  $nodes \leftarrow$ **PrimitiveExpansion**($n_{current}$)
  for n_i *in* $nodes$ **do**
    if $\neg\mathcal{C}$.**contain**(n_i) $\cap$ **CollisionChecking**(n_i) **then**
      $g_{temp} \leftarrow n_c.g_{current} +$ **PrimitiveCost**(n_i);
      if $\neg\mathcal{O}$.**contain**(n_i) **then**
        $\mathcal{O}$.**add**(n_i);
      else if $g_{temp} \geq n_i.g_{current}$ **then**
        continue;
      $n_i.parent \leftarrow n_{current}, n_i.g_{current} \leftarrow g_{temp}$;
      $n_i.f_{current} \leftarrow n_i.g_{current} +$ **HeuristicCost**(n_i);

3.3.1 Mobility-based primitive expansion

UGR's mobility should be discussed before primitive expansion, as it determines UGR's capacity to execute the movement. UGRs can be generally categorized into omnidirectional, all-steering, corner-steering, skid-steering and Ackerman-steering. The omnidirectional system is the most capable, whose holonomic wheels can make instantaneous translation in any direction (Figure 4. a). While restricted by nonholonomic constraints, the all-steering system composed of steering wheels also allows any direction's translation (Figure 4. b). With the outstanding mobility for following any shaped path, omnidirectional and all-steering systems are grouped into "**Unlimited UGR**" system, which adopt straight primitives of conventional A* algorithm (Figure 4. c) to visit the centers of eight neighborhoods. In addition, a set of quadrant rotations should also be added into each expansion (0°, 45°, 90°, 135°, 180°, 225°, 270°, 315°) to represent unlimited UGR's self-rotation capacity (Figure 5 a).



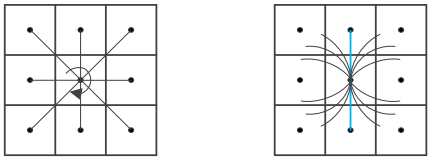
(a) Omnidirectional (b) All-wheel steering

(c) Ackerman-steering (d) Corner-steering (e) Skid-steering

Figure 4. The mobility system of the UGVs

The linear velocity of Ackerman-steering, corner-steering, skid-steering, and is constrained to lie along the forward x-axis of the body (Fig 4, c-e). Hence, they are grouped into "**Limited UGR**" system, whose feasible primitive sets are curves generated based on their steering angles, while the straight extension is a special case of zero steering angle (Figure 5 b). Both skid steering and corner steering make turns by steering

wheels with a maximum steering angle $\boldsymbol{\alpha_{max}}$, while the latter generally has a better steering ability due to the smaller distance from the pivot **P** to the front axle. To keep the balance between efficiency and accuracy, the choice set of these two UGVs' radiuses are determined by dividing the steering range into six parts (Formula 9). Skid-steering UGV's turning is supported by the speed difference between two-side wheels, freeing itself from the turning radius limitation. Similarly, we used a uniform distributed set [±20°, ±40°, ±80°] to represent its outstanding steering ability.



(a) Primitive for limited UGVs (b) Primitive for unlimited UGVs

Fig 5. Primitive of the UGVs.

As a result, new nodes will be generated at the end of each primitive, and the 2D map (x, y) will be extended into the 6D map (X, Y, x, y, θ, m) by additionally recording the world coordinates (**X, Y**), heading $\boldsymbol{\theta}$ and move direction $\boldsymbol{m}$. The new world coordinates of "**Unlimited UGR**" could be easily calculated by converting the cell coordinates, while that of "**Limited UGR**" should be updated based on the drive distance $\boldsymbol{d}$ and max steering angle $\boldsymbol{\alpha_{max}}$ (Formula 1-4). Besides, UGR's heading can be updated by adding rotation degrees to the former heading status, while m is denoted as +1 for forwarding, and -1 for reversing.

$$X' = \begin{cases} X + d\sin\theta, straight \\ X + r\left(\cos\theta - \cos\left(\theta + \frac{180d}{\pi r}\right)\right), curved \end{cases} \quad (1)$$

$$Y' = \begin{cases} Y + d\cos\theta, straight \\ Y + r\left(\sin\left(\theta + \frac{180d}{\pi r}\right) - \sin\theta\right), curved \end{cases} \quad (2)$$

$$\theta' = \begin{cases} \theta, straight \\ \theta + \frac{180d}{\pi r}, curved \end{cases} \quad (3)$$

$$r \in \begin{cases} \left[\pm\frac{L}{\tan\alpha_{max}}, \pm\frac{L}{\tan\frac{\alpha_{max}}{2}}, \pm\frac{L}{\tan\frac{\alpha_{max}}{4}}\right], Ackerman-steering \\ \left[\pm\frac{L}{2*\tan\alpha_{max}}, \pm\frac{L}{2*\tan\frac{\alpha_{max}}{2}}, \pm\frac{L}{2*\tan\frac{\alpha_{max}}{4}}\right], Corner-steering \\ \left[\pm\frac{L}{\tan 80°}, \pm\frac{L}{\tan 40°}, \pm\frac{L}{\tan 20°}\right], Skid-steering \end{cases} \quad (4)$$

3.3.2 Collision checking of the primitive

The moving object is usually considered as a particle in traditional A*, which ignores its actual shape. However, this method may lead to the failure of primitive extension in indoor environments, where corridors are highly likely to be narrower than the robot length (e.g. case in Figure 6). To ensure both the primitive feasibility and the success rate of path planning, a three-stage ***CollisionChecking()*** is proposed, including "Axis-Aligned Bounding Boxes Intersection" (AABBI)checking, "Line-Line Intersection" (LLI) checking, and "Point-in-Rectangle Intersection" (PRI) checking. The basic idea is to detect if the rectangle covered by the robot is intersected with the rectangles covered by other obstacles, and we use the world coordinate for the following calculations.

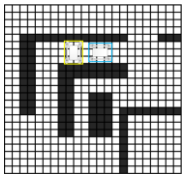
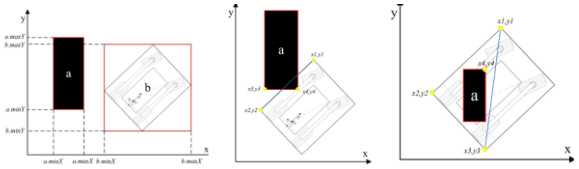


Figure 6. Robots in a congested corridor

AABBI checking is a fast method to test if two no-rotation rectangles intersect in 2D space, and intersection happens if their corners satisfy Equation (5) and (6). Since either obstacle or robot's bounding box is not always axis-aligned, AABBI checking here serves as a speed-up process. The collision between rectangles will not exist if their bounding boxes are not intersected with each other (as shown in Figure 7. a).

$$a.minX \geq b.maxX \;||\; a.maxX \leq b.minX \quad (5)$$

$$a.minY \geq b.maxY \;||\; a.maxY \leq b.minY \quad (6)$$



(a) AABBI (b) LLI (c) PMR

Figure 7. Three-stage collision checking

Secondly, LLI checking should be executed for better accuracy, each line of the robot's rectangle should be checked with all lines of obstacles' rectangles pairwise (Figure 7. b). Two segments (e.g. $[(x_1, y_1), (x_2, y_2)]$, $[(x_3, y_3), (x_4, y_4)]$) meet the requirement of Equation (7-9) are intersected, with the idea that the intersection point should between two endpoints of each segment.

$$(y_4 - y_3)(x_2 - x_1) - (x_4 - x_3)(y_2 - y_1) \neq 0 \quad (7)$$

$$0 \leq \frac{(x_4 - x_3)(y_1 - y_3) - (y_4 - y_3)(x_1 - x_3)}{(y_4 - y_3)(x_2 - x_1) - (x_4 - x_3)(y_2 - y_1)} \leq 1 \quad (8)$$

$$0 \leq \frac{(x_2 - x_1)(y_1 - y_3) - (y_2 - y_1)(x_1 - x_3)}{(y_4 - y_3)(x_2 - x_1) - (x_4 - x_3)(y_2 - y_1)} \leq 1 \quad (9)$$

One special collision case that will escape from LLI checking is that a rectangle is inside another one. As the

final checking step, PRI checking will divide the rectangle of the obstacle into two triangles. Then it will check if any corner of the robot rectangle can be expressed as a linear combination of either triangle's corners, which satisfies Equation (10-12) (Figure 7. c). After repeating for the case that the obstacle is inside the robot, the primitive that passes all the checking is considered as a feasible expansion.

$$0 \leq \frac{(y_2 - y_3)(x - x_3) + (x_3 - x_2)(y - y_3)}{(y_2 - y_3)(x_1 - x_3) + (x_3 - x_2)(y_1 - y_3)} \leq 1 \quad (10)$$

$$0 \leq \frac{(y_3 - y_1)(x - x_3) + (x_1 - x_3)(y - y_3)}{(y_2 - y_3)(x_1 - x_3) + (x_3 - x_2)(y_1 - y_3)} \leq 1 \quad (11)$$

$$0 \leq \frac{x(y_1 - y_2) + x_1(y_2 - y) + x_2(y - y_1)}{(y_2 - y_3)(x_1 - x_3) + (x_3 - x_2)(y_1 - y_3)} \leq 1 \quad (12)$$

3.3.3 Improved cost setting

We aim to find a path that is optimal in energy and time, so the actual cost g of a path is defined as:

$$g = \sum_{n=1}^{N} d(n) + \alpha|\theta(n) - \theta(n-1)| + \beta|s(n) - s(n-1)| + \gamma * m(n-1) \oplus m(n) \quad (13)$$

Specifically, the ***PrimitiveCost()*** is composed of moving distance, orientation difference, steering angels, and motion difference. Thus, a UGR should travel with the least distance $d(n)$, changes of orientation $|\theta(n) - \theta(n-1)|$, changes of steering angle $|s(n) - s(n-1)|$, and switch of motion direction $m(n-1) \oplus m(n)$. α, β and γ are coefficients decided by users. The ***HeuristicCost()*** is measured by Euclidean distance between the robot's current position and goal position as the basic A*.

4 Validation

In the validation section, the fully integrated professional game engine Unity3D is selected to validate the whole path planning framework. Due to the integration of a powerful physical engine PhysX, Unity3D has shown great performance on the simulation of kinematics and dynamics. Particularly, previous studies [13] have proven Unity3D's ability to validate the efficiency of robot path planning algorithms. In addition, as Unity3D offers us significant flexibility to simulate different environments and robot mobility systems, it could be highly time-saving and economical.

4.1 Experiment setting

The fifth floor of Cheng Yu Tung Building, a laboratory and office building of Hong Kong University of Science and Technology, is selected as the environment (shown in Figure 8 with its BIM model). It is a representative place for robot applications, where many tedious works (e.g. material handling, cleaning) could be taken by robots.

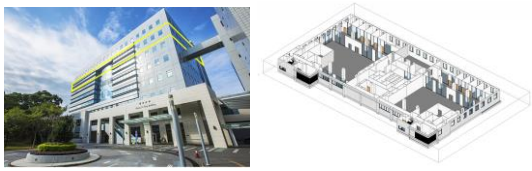


Figure 8. Cheng Yu Tung Building of HKUST

While five mobility system has been discussed, an Ackerman-steering transport robot is herein selected for validation, which is a very common method to delivery objects in indoor areas (shown in Figure 9). And the starting point and ending point are selected in a laboratory (Figure 10).

Figure 9. Ackerman-steering transport robot

Figure 10. Start and goal of the path

4.2 Results

4.2.1 IFC-based integration and mapping

Following the method in section 3.1, the robot model and its relevant information is integrated into BIM, and an overview of its BIM model is shown in Figure 10. After extracting the necessary information from IFC files, a layered grid map is constructed (Figure 11), where the robot's safety buffer (blue) and dynamic update of *Ifcdoor* (yellow rectangle) are included. All space is considered public in this study, and grey ground/cells indicate the free space.

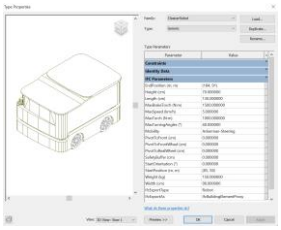


Figure 10. IFC model of the transport Robot

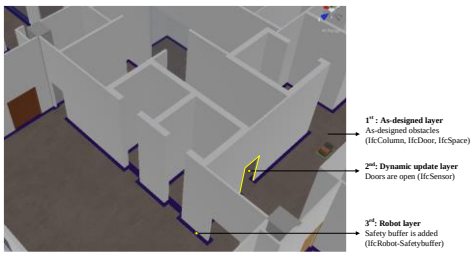


Figure 11. IFC-based indoor map

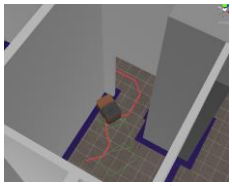
4.2.2 Performance of adaptive A* path planning

To evaluate the performance of adaptive A* algorithm, IFC information of the robot is extracted to develop the robot model in Unity3D. Its physical properties and kinematics are all considered to guarantee the fidelity of the robot. In addition, a PID controller is designed to simulate its movements.

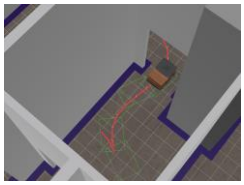
Tradition A* and its integration with our improvements are selected for comparison. According to metrics, computation time is used to test how efficient the algorithm is, while steering jerk and speed jerk are used to test the energy-efficient performance. Trajectory time and success rate are the other two metrics that are commonly considered. We run 20 runs for each scenario and the average values are used for final evaluation. As shown in Table 2, our adaptive A* outperforms others in all the metrics except the computation time, which is expected as more calculations are executed during the path planning process. The demonstration of the failure case of adaptive A* without kinematic primitive and success case of adaptive A* is provided in Figure 12.

Table 2. IFC-based robot properties

	Comp. Time (s)	Steering Jerk (rad/0.2s^2)	Speed Jerk (m/0.2s^2)	Traj. Time (s)	Succ. Rate(%)
Ours without kinematic primitive	/	/	/	/	0
Ours without collision checking	/	/	/	/	0
Ours without improved Cost	2.031	1926.38	740.5	114.4	100%
Adaptive A* (ours)	2.703	1466.66	707.35	97.72	100%



(a) Failure case of adaptive A* without kinematic primitive



(b) Success case of adaptive A*

Figure 12. Path planning progress

5 Conclusion

This paper presents a BIM-integrated framework for unmanned ground robot. Integrating robot into BIM results in a useful knowledge base for UGR's indoor path planning. In addition, an IFC-based layered method is proposed to establish the relationship between BIM and indoor map, followed by a mobility-based A* algorithm that adaptive to different kinds of UGRs.

However, there still exist some limitations to be improved in the future. Firstly, this study uses an Ackerman-steering robot for the validation but does not have a demonstration for unlimited UGRs. The future experiments will try to cover this part. In addition, only a laboratory scenario is considered during path planning, while scenarios with different complexities should be explored in the future.

Acknowledgement

The authors would like to thank "Key-Area Research and Development Program of Guangdong Province" for providing partial support to this research.

References

[1] S. Aine, S. Swaminathan, V. Narayanan, V. Hwang, M. Likhachev, Multi-heuristic a, The International Journal of Robotics Research 35 (1-3) (2016) 224-243.

[2] P. Boguslawski, L. Mahdjoubi, V. Zverovich, F. Fadli, Automated construction of variable density navigable networks in a 3D indoor environment for emergency response, Automation in Construction 72 (2016) 115-128.

[3] G.M. Bone, S.G. Olsen, G.E. Ashby, Autonomous robotic dozing for rapid material removal, ISARC. Proceedings of the International Symposium on Automation and Robotics in Construction, Vol. 30, IAARC Publications, 2013, p. 1.

[4] G. Brown, C. Nagel, S. Zlatanova, T.H. Kolbe, Modelling 3D topographic space against indoor navigation requirements, Progress and

new trends in 3D geoinformation sciences, Springer, 2013, pp. 1-22.

[5] J.C. Cheng, K. Chen, P.K.-Y. Wong, W. Chen, C.T. Li, Graph-based network generation and CCTV processing techniques for fire evacuation, Building Research & Information 49 (2) (2021) 179-196.

[6] D. Dolgov, S. Thrun, M. Montemerlo, J. Diebel, Path planning for autonomous vehicles in unknown semi-structured environments, The International Journal of Robotics Research 29 (5) (2010) 485-501.

[7] T. Fan, P. Long, W. Liu, J. Pan, Distributed multi-robot collision avoidance via deep reinforcement learning for navigation in complex scenarios, The International Journal of Robotics Research 39 (7) (2020) 856-892.

[8] C. Follini, V. Magnago, K. Freitag, M. Terzer, C. Marcher, M. Riedl, A. Giusti, D.T. Matt, BIM-Integrated Collaborative Robotics for Application in Building Construction and Maintenance, Robotics 10 (1) (2021) 2.

[9] A. Hamieh, A.B. Makhlouf, B. Louhichi, D. Deneux, A BIM-based method to plan indoor paths, Automation in Construction 113 (2020) 103120.

[10] P.E. Hart, N.J. Nilsson, B. Raphael, A formal basis for the heuristic determination of minimum cost paths, IEEE transactions on Systems Science and Cybernetics 4 (2) (1968) 100-107.

[11] I. Jeong, Y. Jang, J. Park, Y.K. Cho, Motion Planning of Mobile Robots for Autonomous Navigation on Uneven Ground Surfaces, Journal of Computing in Civil Engineering 35 (3) (2021) 04021001.

[12] J.-Y. Jun, J.-P. Saut, F. Benamar, Pose estimation-based path planning for a tracked mobile robot traversing uneven terrains, Robotics and Autonomous Systems 75 (2016) 325-339.

[13] K.H. Kim, S. Sin, W. Lee, Exploring 3D shortest distance using A* algorithm in Unity3D, TechArt: Journal of Arts and Imaging Science 2 (3) (2015) 1-5.

[14] P. Kim, J. Park, Y.K. Cho, J. Kang, UAV-assisted autonomous mobile robot navigation for as-is 3D data collection and registration in cluttered environments, Automation in Construction 106 (2019) 102918.

[15] N. Li, B. Becerik-Gerber, B. Krishnamachari, L. Soibelman, A BIM centered indoor localization algorithm to support building fire emergency response operations, Automation in Construction 42 (2014) 78-89.

[16] J.J.-C. Lin, W.-H. Hung, S.-C. Kang, Motion planning and coordination for mobile construction machinery, Journal of Computing in Civil Engineering 29 (6) (2015) 04014082.

[17] Y.-H. Lin, Y.-S. Liu, G. Gao, X.-G. Han, C.-Y. Lai, M. Gu, The IFC-based path planning for 3D indoor spaces, Advanced Engineering Informatics 27 (2) (2013) 189-205.

[18] D.V. Lu, D. Hershberger, W.D. Smart, Layered costmaps for context-sensitive navigation, 2014 IEEE/RSJ International Conference on Intelligent Robots and Systems, IEEE, 2014, pp. 709-715.

[19] I. Pohl, Heuristic search viewed as path finding in a graph, Artificial intelligence 1 (3-4) (1970) 193-204.

[20] C. Richter, A. Bry, N. Roy, Polynomial trajectory planning for aggressive quadrotor flight in dense indoor environments, Robotics research, Springer, 2016, pp. 649-666.

[21] C. Schlette, J. Roßmann, Sampling-based floor plan analysis on BIMs, ISARC. Proceedings of the International Symposium on Automation and Robotics in Construction, Vol. 33, IAARC Publications, 2016, p. 1.

[22] H. Shi, L. Shi, M. Xu, K.-S. Hwang, End-to-end navigation strategy with deep reinforcement learning for mobile robots, IEEE Transactions on Industrial Informatics 16 (4) (2019) 2393-2402.

[23] A.R. Soltani, H. Tawfik, J.Y. Goulermas, T. Fernando, Path planning in construction sites: performance evaluation of the Dijkstra, A∗, and GA search algorithms, Advanced Engineering Informatics 16 (4) (2002) 291-303.

[24] Y. Tan, Y. Song, X. Liu, X. Wang, J.C. Cheng, A BIM-based framework for lift planning in topsides disassembly of offshore oil and gas platforms, Automation in Construction 79 (2017) 19-30.

[25] B. Wang, H. Li, Y. Rezgui, A. Bradley, H.N. Ong, BIM based virtual environment for fire emergency evacuation, The Scientific World Journal 2014 (2014).

[26] M. Xu, S. Wei, S. Zlatanova, R. Zhang, BIM-BASED INDOOR PATH PLANNING CONSIDERING OBSTACLES, ISPRS Annals of Photogrammetry, Remote Sensing & Spatial Information Sciences 4 (2017).

[27] B. Zhou, F. Gao, L. Wang, C. Liu, S. Shen, Robust and efficient quadrotor trajectory generation for fast autonomous flight, IEEE Robotics and Automation Letters 4 (4) (2019) 3529-3536.