An Innovative Collision Detection Algorithm for Fast Crane-lift Path Planning in High-rise Modular Integrated Construction

Aimin Zhu, Wei Pan
The University of Hong Kong, China
zhuaimin@connect.hku.hk

Abstract. Crane-lift path planning is crucial for achieving safe and efficient module hoisting in high-rise modular integrated construction (MiC) due to its high demand for crane-lifting. However, most of the existing crane-lift pathfinding methods are time-consuming because of inefficient collision detection mechanisms. This paper aims to develop a novel collision detection algorithm that enables fast path planning while maintaining continuous obstacle detection. An octree and oriented bounding box (Oct-OBB) integrated algorithm is designed with a continuous collision detection strategy. This algorithm includes two components: an optimized octree to uniformly divide the workspace and an OBB algorithm to perform collision detection. The Oct-OBB algorithm was evaluated using a real-life MiC project. The results show that the algorithm successfully generated collision-free paths with only one second, far faster compared to 21 seconds using conventional methods. The developed approach well addresses both the temporal efficiency and spatial continuity of collision detection in crane lifts.

1. Introduction

Automatic crane-lift path planning has become a popular technique for improving both safety and efficiency during module installation in high-rise modular integrated construction (MiC). Examples include using particle swarm optimization and simulated annealing (PSO-SA) integrated algorithm for tower crane path planning (Zhu et al., 2022). Nevertheless, it is important to note that path planning algorithms alone are not sufficient for guaranteeing collision-free and time-efficient crane-lift paths. To achieve these goals, efficient collision detection strategies should also be employed for avoiding the lifted payload colliding with surrounding obstacles (Dutta et al., 2020).

Bounding volume hierarchy (BVH) and space partitioning related techniques are commonly found in the literature for implementing fast collision detection. For example, Hu et al. (2021) utilized an axis-aligned bounding box (AABB) collision detection algorithm for speeding up mobile crane path planning, whereas each iteration required traversing all obstacles. To reduce the path planning time, some researchers developed space partitioning algorithms that conduct collision detection by checking only the surrounding obstacles, while lifted payloads were typically simplified as points (Zhou et al., 2021). Such simplification sacrificed the accuracy in checking for intersections between obstacles and payloads, e.g., the volumetric modules largely possible occurred in several subspaces.

To address the knowledge gap in the previous studies, this paper aims to develop an efficient collision detection algorithm that enables fast path planning while maintaining continuous obstacle detection in high-rise MiC projects. An octree and oriented bounding box (Oct-OBB) integrated algorithm was developed. This algorithm consisted of two components: (1) an optimized octree to uniformly divide the lifting space thereby finding the surrounding obstacles, and (2) an OBB algorithm to perform specific collision detection. An adaptive variable-step strategy was designed by extending the configurations of the planned path to ensure continuous collision detection. The Oct-OBB algorithm was evaluated using a real-life high-rise MiC

project. The selection of the case project adopted the purposive sampling method based on two considerations. One is that there was heavy lifting in the case project. The project involved the crane-lifting of 3824 modules of four 16-story and one 17-story blocks, which is considered significant. The other is that permission was provided to the researchers to access the project data for study.

The remaining of this paper is as follows. Section 2 reviews collision detection issues in crane-lift path planning. Section 3 introduces the research methodology. Section 4 evaluates the proposed algorithm using a real-life MiC project. Section 5 discusses the effectiveness and efficiency of the algorithm, followed by Section 6 that provides the conclusions of the paper.

2. Literature review

2.1 Overview of collision detection

The problem of collision detection involves identifying whether one or more pairs of objects occupy the same space at the same time within a given time frame (Dinas and Bañón, 2015). Pairs of objects are typically represented using various geometric shapes such as spheres and pentahedrons, which consist of vertices, edges, and facets (Bergen, 2003).

Collision detection has been widely adopted in manufacturing, robotics, and computer graphics, such as finding a collision avoidance path for robot motion planning (Elbanhawi and Simic, 2014). However, few studies considered collision detection in the construction industry due to the long construction time and complex workspace. For example, Lai and Kang (2009) proposed a collision detection algorithm to improve the speed of rendering virtual construction environments, but it may be less effective in certain scenarios such as high-rise building projects. Furthermore, some researchers have utilized bounding box representations of on-site mobile machineries (e.g., excavators) and conducted collision checks within a BIM environment to ensure safe movement (Lin et al., 2015).

Regarding collision detection in crane lifts, researchers have given priority to the applications of crane-lift path planning algorithms such as rapidly-exploring random trees in static construction environment (Hu et al., 2021; Zhang and Hammad, 2012), while the research for fast crane-lift collision detection was still at a nascent stage, particularly for high-rise MiC projects (Zhang and Pan, 2020; Pan and Hon, 2020). In fact, collision detection has become a stumbling block to implementing real-time crane-lift path planning in complex construction environments. Relevant collision detection algorithms in crane-lifts are reviewed in Section 2.2.

2.2 Collision detection algorithms for crane-lift path planning

Multiple methods have been proposed for dealing with the collision detection problem including BVH and space partitioning (Klosowski et al., 1998). BVH subdivides target objects into simplified shapes such as bounding spheres to minimize collision detection time, whereas space partitioning separates the three-dimensional (3D) space into primitives or cells to reduce the working range.

These two types of collision detection algorithms are partially employed in lifting path planning. Some researchers have used simplified bounding volumes to replace the complex original objects. For example, Zhang and Hammad (2012) and AlBahnassi and Hammad (2012) developed simplified mobile crane models using bounding boxes. By contrast, a continuous and discrete collision detection engine was built using an OBB algorithm in crane-lifts (Dutta et al.,

2020). Furthermore, Olearczyk et al. (2014) proposed a Weiler–Atherton algorithm to quickly derive a high-quality crane-lift path.

Furthermore, some research teams have adopted space partitioning techniques to accelerate finding a collision-free lifting path. For example, Taghaddos et al. (2018) decomposed concave polygons into the convex ones using a slab decomposition technique. Likewise, Zhou et al. (2021) proposed a cell approach that divided the configuration space into identical cells, where collision detection was performed exclusively in the nearest cell to the lifted payload.

However, current crane-lift collision detection algorithms cannot achieve time-space coherence, particularly in the context of high-rise MiC. Typically, BVH-based algorithms necessitate the traversal of all obstacles in the crane-lift workspace thereby increasing the pathfinding time (Hu et al., 2021; AlBahnassi and Hammad 2012), whereas methods related to space partitioning commonly simplify construction objects as spatially distributed points, resulting in reduced checking accuracy (Zhou et al., 2021).

To fully leverage the benefits of both kinds of collision detection algorithms, an octree and OBB integrated algorithm was developed considering the tradeoffs between temporal efficiency and spatial continuity.

3. Methodology

3.1 Research design

The overview of the research design is provided in Figure 1. First, a comprehensive literature review was conducted to examine the collision detection techniques, which confirmed a shortage of efficient collision detection algorithms in crane-lift path planning, particularly in high-rise MiC projects. Second, an octree and OBB integrated algorithm was developed, and an adaptive variable-step strategy was proposed to ensure continuous collision detection. Third, a real-life high-rise MiC project was used to evaluate the effectiveness and efficiency of the developed algorithm. The algorithm was compared with conventional OBB algorithm to validate the performance improvements achieved. Based on the feedback received from the case project, an optimized collision detection model was determined.

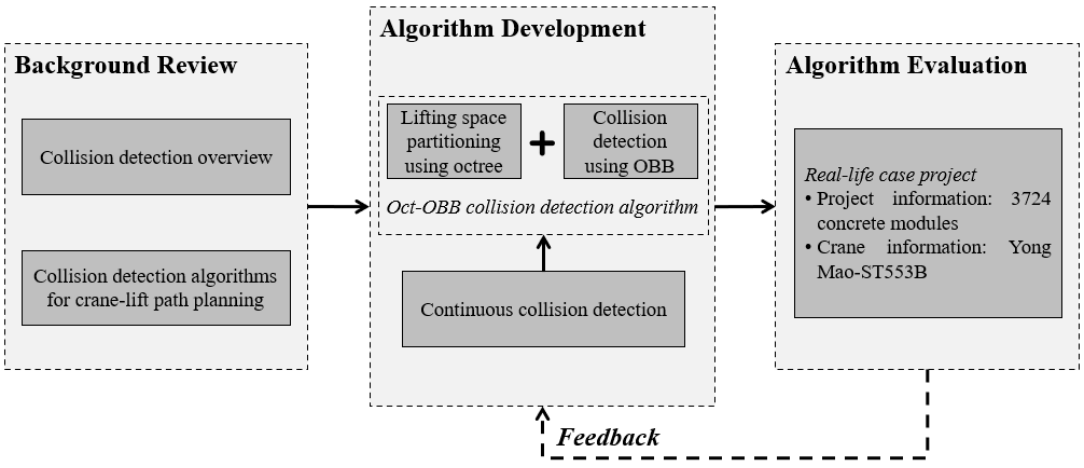
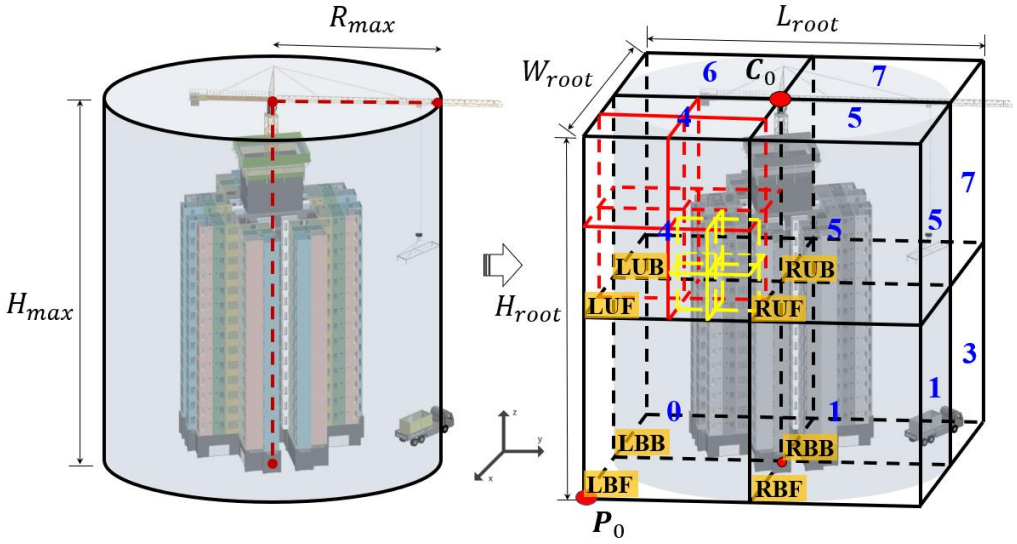


Figure 1: Overview of the research design

3.2 Lifting space partitioning using octree

An octree is a hierarchical tree structure used to partition the target space recursively thereby reducing the search range. Octree has been widely used in various research fields such as detecting collisions between two adjacent robot arms (Shaffer and Herb, 1992).

Figure 2 demonstrates the decomposition of a lifting space into an octree. To minimize the search range, the working boundary is formed as a cylinder (Figure 2 (a)), where the maximum working height, H_{max}, is equivalent to the height of the slewing center, and the maximum working radius, R_{max}, is determined by the working range between the start point and end point of the lifted payload. To include all the objects in the lifting space, the root leaf node is derived by an external rectangle of the cylinder, as shown in Figure 2 (b). Each father leaf node will generate precisely 8 child leaf nodes during the space partitioning process, and they are located at Right-Bottom-Front (RBF), Left-Bottom-Front (LBF), Left-Bottom-Back (LBB), Right-Bottom-Back (RBB), Right-Up-Front (RUF), Left-Up-Front (LUF), Left-Up-Back (LUB), and Right-Up-Back (RUB) of the father node, respectively. Therefore, if the crane slewing center is $\boldsymbol{C_0} = (x_0, y_0, z_0)$, the LBF vertex coordinate of the root node is ruled by $\boldsymbol{P_0} = (x_0 - \frac{L_{root}}{2}, y_0 - \frac{L_{root}}{2}, z_0 - H_{root})$. Furthermore, another important concept is the depth of an octree, denoted as d, referring to the number of times the root node divided into sub-nodes.



(a) Working boundary definition (b) Octree generation

Figure 2: Lifting space partitions using octree

To localize the leaf nodes and present collision detection in the octree, the dimensions and coordinates of leaf nodes should be determined.

The dimension of a leaf node at a *d*-depth octree is written as

$$\boldsymbol{D} = \begin{bmatrix} L_{root} & 0 & 0 \\ 0 & W_{root} & 0 \\ 0 & 0 & H_{root} \end{bmatrix} \cdot 2^{-d} \qquad (1)$$

where $\boldsymbol{D}$ is the dimension matrix; $L_{root} = 2R_{max}$; $W_{root} = 2R_{max}$; $H_{root} = H_{max}$.

In addition, the vertex coordinate of each leaf node is determined based on its LBF coordinate, and the LBF coordinate matrix of every 8 child nodes is calculated as

$$\boldsymbol{LBF} = \begin{bmatrix} x_l & x_l + \frac{L}{2} & x_l & x_l + \frac{L}{2} & x_l & x_l + \frac{L}{2} & x_l & x_l + \frac{L}{2} \\ y_l & y_l & y_l + \frac{W}{2} & y_l + \frac{W}{2} & y_l & y_l & y_l + \frac{W}{2} & y_l + \frac{W}{2} \\ z_l & z_l & z_l & z_l & z_l + \frac{H}{2} & z_l + \frac{H}{2} & z_l + \frac{H}{2} & z_l + \frac{H}{2} \end{bmatrix} \quad (2)$$

where the elements from columns 1 to 8 of matrix $\boldsymbol{LBF}$ correspond to the vertex coordinates of LBF leaf nodes. Since the branch order of each leaf node is fixed, it is possible to obtain the left seven vertex coordinates of every leaf node based on the $\boldsymbol{LBF}$ matrix. In addition, L, W, and H represent the length, width and height of the target leaf node, respectively.

3.3 Collision detection using OBB

The idea of detecting obstacles in crane-lifts using the OBB algorithm is to simplify the lifted payloads and obstacles as bounding boxes and check intersections between them. In this paper, an OBB algorithm was used, which has two functions: (1) identifying the number of obstacles in each leaf node of the octree, and (2) presenting collision detection between the planned path and its surrounding obstacles.

The octree separation process stops when the number of obstacles in each leaf node falls below the threshold value. This approach ensures that the space is segmented uniformly, which in turn allows for quick localization of any point to its corresponding leaf node. Following OBB collision detection, the obstacles present in each leaf node are recorded.

PSO-SA algorithm, a nature-inspired metaheuristic, was developed and used to find the optimized crane-lift paths in high-rise MiC projects (Zhu et al., 2022). However, it would take approximately one minute to search for a high-quality solution space, because of the traversal of all obstacles in the lifting space. Instead, by searching the leaf nodes surrounded by the planned path, it is easy to screen out the surrounding obstacles of the planned path. Therefore, a fast path planning process can be achievable by presenting collision detection between the planned path and its surrounding obstacles.

3.4 A continuous collision detection strategy

To enable continuous collision detection in the lifting space, an adaptive variable-step strategy was developed. This strategy involves incorporating new configurations into the existing planned path. The reason for this is that the initial configurations in the path are spatially sparse, which means that although they may not collide, collisions may still occur at other locations along the planned path.

The configuration set in a lifting path is written as $\boldsymbol{X}$. As a result, the i^{th} configuration in one of the planned paths is denoted as $\boldsymbol{X^i} = (X_{i,1}, X_{i,2}, X_{i,3}, X_{i,4})$, where $X_{i,1}, X_{i,2}, X_{i,3}, X_{i,4}$ correspond to self-rotation of the hook (i.e., "Rotation"), jib slewing (i.e., "Slewing"), trolley movement (i.e., "Movement") and sling hoisting (i.e., "Hoisting"), respectively; $i \in [1, N]$, and N denotes the number of configurations in the planned path.

New configurations are generated based on the changes in DOFs between two adjacent configurations. In order to ensure that all parts of the path can be detected, it is essential to limit the maximum movement distance of each module. This distance should not exceed the

minimum geometric size in the direction of its motion (Figure 3). The configuration step in the actuated DOFs of the crane is written separately as

$$n_{jb} = \left\lfloor \frac{\left(X_{i+1,2} - X_{i,2}\right) \times X_{i,3}}{\Delta L_{jb}} \right\rfloor + 1 \tag{3}$$

$$n_{tr} = \left\lfloor \frac{X_{i+1,3} - X_{i,3}}{\Delta L_{tr}} \right\rfloor + 1 \tag{4}$$

$$n_{sl} = \left\lfloor \frac{X_{i+1,4} - X_{i,4}}{\Delta L_{sl}} \right\rfloor + 1 \tag{5}$$

where, (1) n_{jb} denotes the number of new configurations in the aspect of "Slewing", $\Delta L_{jb} = \overline{AB} \approx \widetilde{AB}$ (i.e., ΔL_{jb} approximately equivalent to the arc length of AB), $\Delta\theta \approx \frac{\Delta L_{jb}}{R} = \frac{\Delta L_{jb}}{X_{i,3}}$, $\theta = \left(X_{i+1,2} - X_{i,2}\right)$, $\max \Delta L_{jb} = \min\{W_m, L_m\}$; (2) n_{tr} denotes the number of new configurations in the aspect of "Movement", $L_{tr} = X_{i+1,3} - X_{i,3}$, $\max \Delta L_{tr} = \min\{W_m, L_m\}$; and (3) n_{sl} denotes the number of new configurations in the aspect of "Hoisting", $L_{sl} = X_{i+1,4} - X_{i,4}$, $\max \Delta L_{sl} = H_m$. These three configuration steps are determined by rounding down Equations (3)-(5).

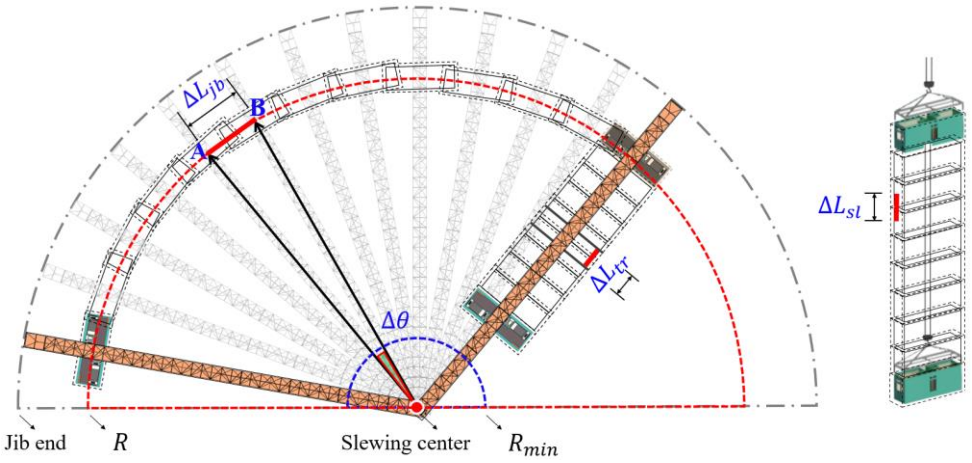


Figure 3: Process of continuous collision detection

4. Results and analyses of the case study

The effectiveness and efficiency of the proposed Oct-OBB collision detection algorithm was evaluated using a real-life high-rise MiC project. The project consisted of a total of five blocks which were constructed using 3724 precast concrete modules to form 648 flat units. One block was of 17 stories, while the remaining four blocks were of 16 stories each. Constructing each typical floor required precise installation of 44 precast concrete modules within a floor cycle of five days. This demanding schedule necessitated meticulous planning and coordination to guarantee timely delivery and installation of each module. The mode of Yong Mao-ST553B was used as the tested crane.

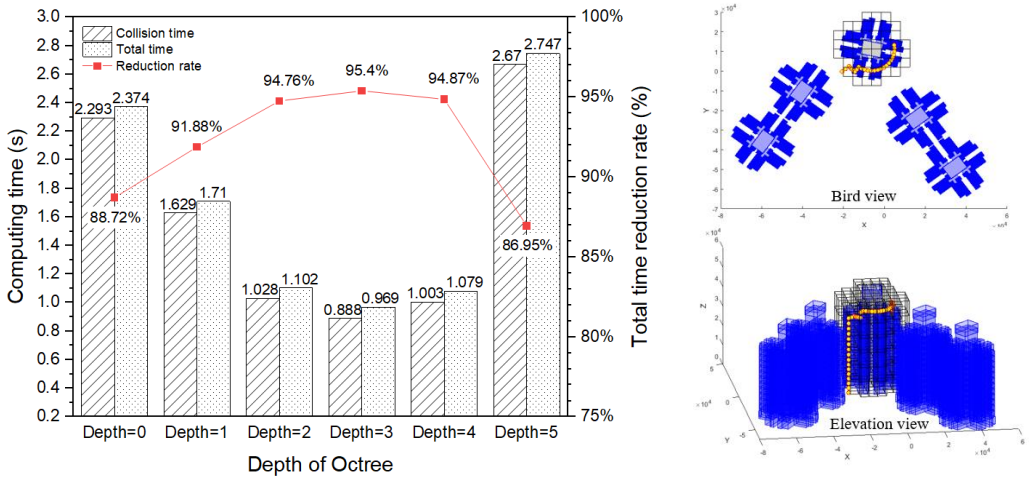
As is shown in Table 1, it took approximately 21s to find an optimized lifting path using only OBB algorithm for path planning. However, this method is more time-consuming for complex scenarios, such as high-rise modular construction projects involving multiple building blocks (Pan and Hon, 2020).

Table 1: Results of collision detection and octree structure under various depths of octree

Collision detection algorithm	Depth of octree	No. of leaf nodes	Threshold obstacles per leaf	Surrounding obstacles in the planned path	Collision computing time	Total computing time
OBB	-	-	-	-	20.881	21.048
Oct-OBB	0	1	548	548	2.293	2.374
	1	8	156	255	1.629	1.71
	2	52	57	145	1.028	1.102
	3	300	21	62	0.888	0.969
	4	1677	10	22	1.003	1.079
	5	10985	6	19	2.67	2.747

Note: total number of obstacles in the lifting space is 2744

To further reduce the computing time, an innovative octree structure was used to partition the lifting space and to select the surrounding obstacles. Table 1 provides the results of collision detection and octree structure under various octree depths. As compared to the global traversal method that checks collisions between the lifted module and all obstacles in the lifting space, the use of the octree structure led to a reduction in computing time of more than 86% (Figure 4 (a)).



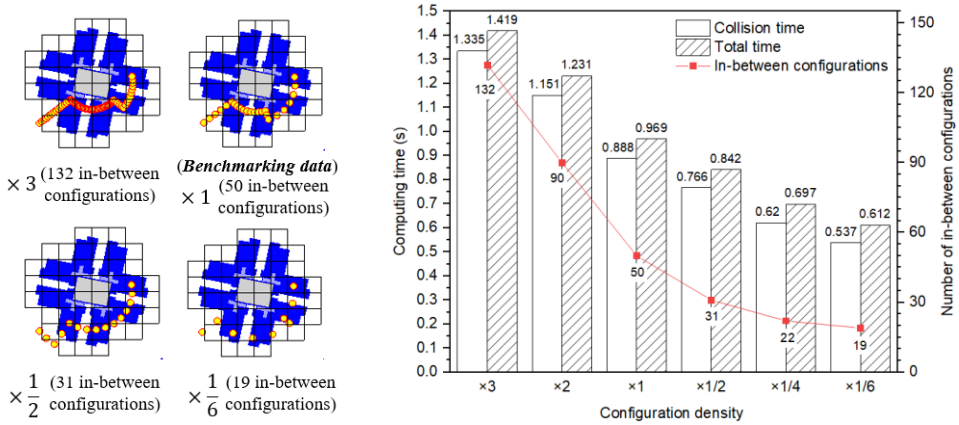
(a) Computing time under various depths of octree (b) 3-depth octree

Figure 4: Results of collision detection under various depths of octree

The most time-efficient situation for crane-lift path planning was at a 3-depth octree, with a 95% reduction in time compared to the non-space partitioning method (Figure 4). Leaf nodes stored obstacles in the lifting space, and collision detection involved determining surrounding leaf nodes and checking their stored obstacles. Small tree depth resulted in increased collision detection time due to multiple obstacles in large-sized leaf nodes (e.g., 255 at a 1-depth octree), while large tree depth also increased collision detection time due to the vast number of leaves to search (e.g., 10965 at a 5-depth octree). Thus, the optimal balance point was achieved at a 3-depth octree, resulting in the lowest computational cost.

In addition, an adaptive variable-step strategy was designed to improve the accuracy of collision detection by incorporating additional in-between configurations along the path, utilizing Equations (3)-(5). The data obtained from these three Equations was taken as the benchmark, while an additional five sets of data were examined, varying from three times larger to one-sixth smaller than the benchmark. As shown in Figure 5 (a), the availability of a satisfactory configuration density ensured that all crucial components along the planned path could be

detected. For example, the benchmarking data allowed for the creation of a spatially continuous crane-lift path with 50 in-between configurations. In contrast, the planned path contained only 19 loosely spaced configurations when the density was reduced to one-sixth that of the benchmark. However, increasing the configuration density in the planned path would lead to greater computational intensity, as demonstrated by the 46.44% increase in time required from 0.969s to 1.419s when the configuration density was tripled (Figure 5 (b)).



(a) Path visualization (b) Computing time under various configuration densities

Figure 5: Results of collision detection under various configuration densities

Therefore, a 3-depth Oct-OBB structure together with the developed continuous collision detection strategy was suggested for achieving an optimal balance between spatial accuracy and computing efficiency.

5. Discussion

The proposed Oct-OBB algorithm has demonstrated remarkable efficiency by quickly finding a collision-free crane-lift path (i.e., 0.969s) while maintaining continuous obstacle detection.

First, an optimized octree method was utilized to partition the workspace into uniform leaf nodes, enabling efficient collision detection of the planned path with surrounding obstacles. This method eliminated the need to traverse every obstacle present within the workspace, thereby improving the collision detection process. For example, compared to the non-space partitioning method, using a 3-depth octree led to a significant reduction in obstacles from 2744 to only 62. Nevertheless, crane-lift path planning has been an area where space partitioning-related methods have been infrequently utilized. There has been a recent surge of interest in this technique, such as a cell decomposition method by Burkhardt and Sawodny (2021) and a cell method by Zhou et al. (2021). However, the lifted payload was commonly depicted as a simplified mass point in previous studies, such as Zhou et al. (2021) and Hu et al. (2021), which greatly sacrificed the spatial accuracy. For example, the lifted payload may be in several leaf nodes at the same time, whereas the point-mass representation is always in some node.

Second, an OBB algorithm was employed for collision detection between objects. The specific procedures involved: (1) identifying obstacles within each leaf node by checking for intersections between the leaf nodes and space obstacles; (2) locating the leaf nodes that were adjacent to the planned path and utilizing the OBB algorithm to detect collisions between the lifted payloads and the obstacles in the detected leaf nodes. This design followed the philosophy of lightening the collision detection tasks by pre-finishing part of the works at the octree modeling stage, thereby reducing collision detection time.

Third, to address the issue of possible collisions between adjacent configurations, an adaptive variable-step approach was employed to facilitate continuous collision detection. The planned path was modified by incorporating supplementary configurations that corresponded with alterations in the DOFs of the crane. The adoption of this strategy resulted in a higher configuration density in the lifting space, leading to an enhancement in the accuracy of collision detection. To achieve this, additional configurations were computed using Equations (3)-(5), whereby the number of in-between configurations was increased to 50, ensuring that all segments of the planned path were detectable (Figure 5). However, this has not been identified in previous studies, e.g., Hu et al. (2021) and Zhou et al. (2021).

Based on the discussion above, this algorithm is expected to be suited for application in complex construction environments such as high-rise MiC. Additionally, it has the potential to be promoted for use in dynamic lifting scenarios such as crane-lift path re-planning.

6. Conclusions

This paper develops an octree and oriented bounding box (Oct-OBB) integrated algorithm to achieve fast crane-lift path planning for high-rise modular integrated construction (MiC). The effectiveness and efficiency of the Oct-OBB algorithm were evaluated using a real-life MiC project. The major findings and the conclusions of the paper are as follows.

The developed Oct-OBB algorithm consists of two components: (1) an optimized octree to uniformly divide the lifting space thereby finding the surrounding obstacles and (2) an OBB algorithm to perform collision detection. An adaptive variable-step strategy was proposed by adding additional configurations into the planned path to ensure continuous collision detection.

The Oct-OBB algorithm has demonstrated a remarkable improvement over traditional methods such as axis-aligned bounding box or OBB algorithms in terms of time-space coherence. The proposed algorithm achieved an impressive processing speed of just one second for finding the optimized collision-free crane-lift paths, far faster compared to 21 seconds using conventional methods. In parallel, continuous and accurate collision detection was implemented during the pathfinding process.

This paper makes significant contributions to the field of intelligent computing in construction. Theoretically, the study is the first of its kind by integrating space partitioning with bounding volume hierarchy algorithms for addressing collision detection problems. Striking a balance between temporal efficiency and spatial continuity, the knowledge of automatic construction planning is advanced by the developed algorithm. Practically, the presented evidence warrants that the developed algorithm should facilitate safe and efficient module installation in high-rise MiC projects by offering optimized operation suggestions to both crane operators and robotized cranes in near real time.

Nevertheless, the current study has some limitations. First, the study solely relied on *MATLAB* running on Windows 11, and it is important to note that both the hardware and software configurations can significantly impact the time complexity of collision detection. Second, the study did not account for the potential impact of payload swing during lifting, which has the potential to pose safety risks and adversely affect operational efficiency. Future research is recommended to explore different simulation platforms such as *Unity 3D* for visualizing the path planning process and to utilize state-of-the-art payload swing control techniques for crane-lift path re-planning.

Acknowledgments

This work is supported by the Research Impact Fund of the Hong Kong Research Grants Council (Project No. HKU R7027-18). Also acknowledged are the Architectural Services Department of the HKSAR Government, Yau Lee Construction Company Limited for providing data access for the case study, and Dr Zhiqian Zhang for participating in this study.

References

AlBahnassi, H., and Hammad A. (2012). Near real-time motion planning and simulation of cranes in construction: Framework and system architecture. Journal of Computing in Civil Engineering, 26, pp. 54-63.

Bergen, Gino van den. (2003). Collision detection in interactive 3d environments. Morgan Kaufmann Publishers, Hounslow, United Kingdom.

Burkhardt, M., and Sawodny O. (2021). A graph-based path planning algorithm for the control of tower cranes. In: 2021 American Control Conference (ACC), 2021, New Orleans, USA.

Dinas, S., and Bañón J.M. (2015). A literature review of bounding volumes hierarchy focused on collision detection. SciELO Analytics, 17, pp. 49-62.

Dutta, S., Cai, Y., Huang, L., and Zheng J. (2020). Automatic re-planning of lifting paths for robotized tower cranes in dynamic BIM environments. Automation in Construction, 110.

Elbanhawi, M., and Simic M. (2014). Sampling-based robot motion planning: A review. IEEE Access, 2, pp. 56-77.

Hu, S., Fang Y., and Guo H. (2021). A practicality and safety-oriented approach for path planning in crane lifts. Automation in Construction, 127.

Klosowski, J.T., Held, M., Mitchell, J.S.B., Sowizral, H., and Zikan, K. (1998). Efficient collision detection using bounding volume hierarchies of k-DOPs. IEEE Transactions on Visualization and Computer Graphics, 4, pp. 21-36.

Lai, K.C., and Kang, S.C. (2009). Collision detection strategies for virtual construction simulation. Automation in Construction, 18, pp. 724-36.

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

Olearczyk, J., Bouferguène A., Al-Hussein, M., and Hermann, U. (2014). Automating motion trajectory of crane-lifted loads. Automation in Construction, 45, pp. 178-86.

Pan, W., and Hon C.K. (2020). Briefing: Modular integrated construction for high-rise buildings. Proceedings of the Institution of Civil Engineers - Municipal Engineer, 173, pp. 64-68.

Shaffer, C.A., and Herb, G.M. (1992). A real-time robot arm collision avoidance system. IEEE Transactions on Robotics and Automation, 8, pp. 149-160.

Taghaddos, H., Hermann, U., and Abbasi, A. (2018). Automated crane planning and optimization for modular construction. Automation in Construction, 95, pp. 219-32.

Zhang, C., and Hammad A. (2012). Improving lifting motion planning and re-planning of cranes with consideration for safety and efficiency. Advanced Engineering Informatics, 26, pp. 396-410.

Zhang, Z., and Pan, W. (2020). Lift planning and optimization in construction: A thirty-year review. Automation in Construction, 118.

Zhou, Y., Zhang, E., Guo, H., Fang, Y., and Li, H. (2021). Lifting path planning of mobile cranes based on an improved RRT algorithm. Advanced Engineering Informatics, 50.

Zhu, A., Zhang, Z., and Pan, W. (2022). Crane-lift path planning for high-rise modular integrated construction through metaheuristic optimization and virtual prototyping. Automation in Construction, 141.