Since January 2020 Elsevier has created a COVID-19 resource centre with free information in English and Mandarin on the novel coronavirus COVID-19. The COVID-19 resource centre is hosted on Elsevier Connect, the company's public news and information website.

Elsevier hereby grants permission to make all its COVID-19-related research that is available on the COVID-19 resource centre - including this research content - immediately available in PubMed Central and other publicly funded repositories, such as the WHO COVID database with rights for unrestricted research re-use and analyses in any form or by any means with acknowledgement of the original source. These permissions are granted for free by Elsevier for as long as the COVID-19 resource centre remains active.
Collision-free trajectory planning for robotic assembly of lightweight structures

Jiangpeng Shu a,b,c, Wenhao Li a, Yifan Gao a,c

a College of Civil Engineering and Architecture, Zhejiang University, Hangzhou 310058, China
b Centre for Balance Architecture, Zhejiang University, Hangzhou 310058, China
c The Architectural Design & Research Institute of Zhejiang University Co. Ltd, Hangzhou 310028, China

ARTICLE INFO

Keywords:
Robotics
Construction automation
Trajectory optimisation
Obstacle avoidance

ABSTRACT

This research presents a trajectory planning approach for robotic assembly of lightweight structures for COVID-19 healthcare facilities. The prefabricated building components of COVID-19 healthcare facilities have non-negligible volume, where the crux of the scientific question lies in how to incorporate geometry-based collision checks in trajectory planning. This research developed an algorithm that refines the RRT* (Rapidly-exploring Random Tree-Star) algorithm to enable the detour of a planned trajectory based on the geometry of prefabricated components to prevent collisions. Testing of the approach reveals that it has satisfactory collision-avoiding and trajectory-smoothing performance, and is time- and labour-saving compared with the traditional human method. The satisfactory results highlight the practical implication of this research, where robots can replace human labour and contribute to the mitigation of COVID-19 spread on construction sites. The subsequent research will investigate the use of a collaborative robot to screw bolt connections after the components are assembled at locations.

1. Introduction

During the COVID-19 pandemic, a large number of construction organisations halted their activities and their employees were encouraged to work from home [1]. The contractor association statistics indicated that nearly 80.35% and 90% of domestic construction projects were put on hold in 2020 in China and the US respectively [2,3]. This accelerates the ongoing trend toward the implementation of labour-saving production techniques such as robotisation [4]. As Brakman et al. [5] have pointed out, robots do not get biological virus and thus ensure that essential construction tasks are not suspended during pandemics. To carry out the robotisation of conventional construction work, one of the approaches is flexible automation using robots based on trajectory planning [6].

Trajectory planning is one of the fundamental problems in robotics [5]. The capability to plan collision-free trajectories is a precondition for autonomous robotic platforms to manoeuvre in a given environment and timely adjust the rotation angle of each joint to eventually reach a target point [7]. In recent years, trajectory planning for construction tasks has received considerable attention in the scientific community. Terada and Murata [8] utilised the Bucket Brigade algorithm to generate a gradient field for identifying the shortest trajectory in the assembly of building blocks. Davtalab et al. [9] used the Lin-Kernighan algorithm to solve toolpath optimisation problems in the concrete contour crafting process. Kontovourkis et al. [10] adopted the Bidirectional Evolutionary Structural Optimisation algorithm to optimise the infill patterns of clay structure by gradually removing inefficient material and established the nozzle trajectory for additive manufacturing based on the polylines of the optimised infill patterns. Ding et al. [11] developed a deep convolutional neural network for the visual identification of brick patterns and utilised the ABB RobotStudio software for brick assembly trajectory planning. By using ABB RobotStudio, the trajectory can be determined as long as the start and goal (assembly) coordinates of each brick are provided. King et al. [6] developed a Rhinoceros-based digital workflow for robotic tile placement, where a digital image can be computationally decomposed into tile mosaic patterns based on grout lines and ABB RobotStudio is used to generate placement trajectories for each tile.

Although researchers have carried out robotic construction-related works, there has been no mature solution specifically designed to deal with the robotic assembly of lightweight structures for COVID-19 healthcare facilities. Research finds that there are regions experiencing COVID-19 patient surge while having insufficient healthcare capacity. 

https://doi.org/10.1016/j.autcon.2022.104520
Received 30 December 2021; Received in revised form 30 June 2022; Accepted 30 July 2022
Available online 3 August 2022
0926-5805/© 2022 Elsevier B.V. All rights reserved.
for patient care [5]. However, to prevent the spread of the COVID-19 virus, a large number of construction organisations had to suspend their activities amid the pandemic as the construction industry is labour-intensive [1]. Against this background, therobotisation of the assembly of COVID-19 healthcare facilities is an urgent need. This research aims to develop a collision-free trajectory planning approach that can assist in the robotic assembly of COVID-19 healthcare facilities. The rest of the paper is organised as follows: section 2 contains a review of research works in the literature related to trajectory planning; section 3 specifies the scope of the robotised construction work and defines the problems to be addressed in the proposed approach; section 4 presents the system architecture of the developed approach; section 5 tests the developed approach in terms of collision avoidance, trajectory smoothness, trajectory length, and execution time; section 6 discusses the theoretical and practical implications; while section 7 summarises the findings, notes the limitations, and recommends future research directions.

2. Related works

The geometrical trajectory is normally defined in the operating space of robots [10]. Trajectory planning aims at identifying a feasible and optimal geometrical trajectory for a manipulated rigid body between a start coordinate and a goal coordinate through a robot’s operating space [5]. The feasible feature is related to collision avoidance. Trajectories are considered more feasible that bring a robot from a start configuration to a goal configuration in a collision-free manner while obeying the robot’s kinematic constraints [12]. The optimal feature is related to smoothness, where trajectories with less sharp turns are considered more optimal [12]. Previous studies [13,14] have investigated the categorisation of trajectory planning algorithms and suggested that such algorithms can take a classical or advanced approach. The classical approach is used for trajectory planning in a predictable environment, where the obstacles are static and there is no need for real-time re-planning [13]. The advanced approach is used to deal with trajectory planning in an unpredictable environment with dynamic obstacles, which entails making real-time modifications to trajectories so that the dynamic obstacles are avoided [13].

There are three major types of classical approaches: the cell decomposition approach, the probabilistic roadmap approach, the potential field approach, and the circular fields method [14].

1. The cell decomposition approach breaks down the configuration space of the problem into nonoverlapping regions referred to as cells, and derives a connectivity graph to represent the adjacency relationships of the cells [14]. Thereafter a graph-search algorithm can be applied to form the robot’s trajectory, such as the Dijkstra’s algorithm, the Bucket Brigade algorithm, and the Lin-Kernighan Heuristic algorithm [15]. As mentioned in section 1, the Bucket Brigade and Lin-Kernighan algorithms have been used respectively by Terada and Murata [8] and Davtalab et al. [9] to solve trajectory planning problems in the construction sector.

2. The probabilistic roadmap approach consists of a roadmap construction phase and a query phase [16]. In the roadmap construction phase, a navigation mesh in the configuration space of the problem is constructed, where the vertices of the mesh are randomly generated in the free space and are connected to their k-nearest neighbouring vertices such that the connecting edges do not cross any obstacle [13]. In the query phase, the navigation mesh is searched by applying computational geometry structures such as the visibility graph for the shortest trajectory and the Voronoi diagram for the maximum clearance trajectory [14].

3. The potential field approach has been widely used in trajectory planning with static obstacles, where the topological structure of the free space (i.e. a configuration space’s obstacle-free segment) is derived in the form of minimum potential valley and electrostatic potential functions are applied to the goal configuration and each obstacle [17,18]. Thereafter the goal configuration and each obstacle form an attractive force and a repulsive force respectively, and the robot is repelled from colliding with the obstacles and is pulled toward the goal configuration. More recently, researchers [19,20] have attempted to adapt the potential field approach for trajectory planning with dynamic obstacles. However, there is still some room for improvement given that the robot can get trapped in a local minimum implementing the adapted approach [19]. The circular fields method

4. The circular fields method is inspired from the physical laws in electromagnetism, which regards a robot as a moving charged particle and conducts trajectory planning by exerting a reactive force on the particle in a hypothesised electromagnetic field [21]. However, the method relies on a priori information of the positions of the obstacles in order to pre-label the obstacles’ coordinates for trajectory planning [22]. In addition, another drawback of the circular fields method is the missing optimality regarding the path length [21].

The advanced approach can be classified into sampling- and optimisation-based categories [15]. While many sampling- and optimisation-based algorithms have been proposed in the past, each of the categories has its pros and cons which are elaborated in the following.

1. The basic idea of optimisation-based algorithms is to establish a set of cost functions that define some considerations as to what an expected optimum trajectory might look like (e.g., minimal trajectory completion time, obstacle constraints), and then identify the optimum trajectory that can minimise the cost functions [23]. Covariant Hamiltonian Optimisation for Motion Planning (CHOMP), Stochastic Trajectory Optimisation for Motion Planning (STOMP), and Trajectory Optimisation for Motion Planning (TrajOpt) are the seminal optimisation-based algorithms used in trajectory planning [24]. CHOMP [25] and its variants [26–28] create a signed distance field for obstacle detection and utilise covariant gradient descent to minimise cost functions. STOMP [29] starts with the creation of an initial stochastic trajectory and then optimises non-differentiable spatial constraints for the trajectory by sampling a series of noisy trajectories around, which are then combined to minimise cost functions in order to obtain an updated optimal trajectory. CHOMP and STOMP finely discretise the robot’s operating space to reason about obstacles, which could result in prohibitive computational cost and solution time [15]. TrajOpt was proposed by Schulman et al. [30] in an effort to reduce solution time, which formulates trajectory planning as sequential quadratic programming and iteratively attempts to converge cost functions to the minimum. However, one of the concerns with optimisation-based algorithms is that they depend on the minimisation of cost functions, which is not always successful in finding the global minimum and can get trapped in a local minimum [24].

2. The basic idea of sampling-based algorithms is to sample random points in the space and reject points that would overlap the obstacles to form a collision-free trajectory [13]. Tree-based sampling algorithms such as Expansive-spaces Tree (EST) and Rapidly-exploring Random Tree (RRT) are the widely used algorithms in this class [13,14,21]. EST and RRT sample the space and grow a tree from the current configuration of the robot until the goal configuration becomes part of the tree [32]. RRT randomly searches the space to explore the expansion direction of the tree, while EST measures the density of the points sampled in the space and biases its exploration toward parts of the space with the lowest density [32]. To reduce the risk of leading to unproductive tree expansions and speed up the solution rate, Gammell et al. [33] proposed the variant—RRT*—as an improvement to RRT, where the tree growth strategy is modified such that the searching direction is guided toward the goal
configuration. The advantage of RRT* compared with optimisation-based algorithms lies in its inherent asymptotical convergence to the global optimality, where the optimisation-based algorithms have a higher chance to get trapped in a local minimum. The RRT* algorithm does not require prior knowledge of the environment, which runs in real time and is capable of planning robot trajectory in an unpredictable environment with dynamic obstacles. In addition, the RRT* algorithm compares the edge lengths between a waypoint in the trajectory and a number of points randomly sampled in the obstacle-free space before a point can be confirmed and added as a connecting waypoint.

The construction site is an unpredictable environment with dynamic obstacles. In addition, for the construction of a building structure, it also needs to find the global optimality in order to reduce the energy consumption of the robotic manipulator. Given the considerations, in this research, the authors chose the RRT* algorithm to develop a trajectory planning approach that can assist in the robotic assembly of COVID-19 healthcare facilities.

3. Problem definition

To prevent the spread of the COVID-19 virus, many countries have elaborated action plans. The UK and US turned convention centres and indoor stadiums into COVID-19 healthcare facilities [34,35]. The Indian government took over all the private healthcare facilities and converted them to COVID-19 hospitals [36]. China built several healthcare facilities to contain the spread of the COVID-19 virus, including the best-known Leishenshan and Huoshenshan hospitals in Wuhan, China [37]. Taking the Leishenshan hospital as an example, this hospital consists of more than 3000 standard flatpack house units [37]. The standard unit is comprised of prefabricated structural components including beams (short edge), beams (long edge), pillars, purlins, floor (roof) panels, and wall panels (Fig. 1a), and is 6.0 m long, 3.0 m wide, and 2.6 m tall (Fig. 1b).

This research puts forward the robotisation of the construction of COVID-19 healthcare facilities and investigates the scope of the robotisation, considering standard flatpack house units. As documented in the recent research [37], the construction of the Leishenshan hospital involves two steps: 1) standard flatpack house units were first manually assembled using prefabricated components, and then 2) the units were hoisted into designated locations using mobile cranes (Fig. 2). However, the physical characteristics of a robot may not allow it to carry out the complete construction process. A standard unit weighs over 970 kg, which is estimated based on the weights of prefabricated components and the number of each component used in the unit (Fig. 3). The total weight goes beyond the handling limit of the largest industrial robotic manipulator model on the market—ABB IRB 8700 (maximum payload: 500 kg). Therefore, the scope of the robotisation in this research focuses on the first construction step—robotic assembly of a standard flatpack unit with prefabricated components. The traditionally used mobile cranes might be more suitable than robotic manipulators for the second construction step—installation of a standard flatpack units into designated locations, given the suitable lifting capability of mobile cranes. Also, this research focuses on the robotic assembly of prefabricated structural components, which does not involve the installation of mechanical, electrical, and plumbing parts (Fig. 3).

The assembly of a flatpack house unit consists of multiple

![Fig. 1. The standard flatpack house unit used in the Leishenshan hospital: a) the prefabricated structural components—beams (short edge), beams (long edge), pillars, purlins, floor (roof) panels, and wall panels; b) the dimensions of the flatpack house unit.](image)
prefabricated components, which leads to a multi-query version of trajectory planning problems. To solve the multi-query problem, the development of the approach considers a combination of techniques as presented below:

1. Determining the assembly coordinates of prefabricated building components based on the spatial analysis of a flatpack house BIM model.
2. Sequencing the involvement of the prefabricated components in the global assembly planning process based on the spatial analysis of the flatpack house BIM model.
3. Defining trajectories for the assembly of each prefabricated component based on the RRT* algorithm. In the existing studies in the construction field [9–11], the end-tips of robots are commonly represented in a particle setting for trajectory planning. However, a robot may collide with surrounding objects given that a manipulated prefabricated building component has non-neglectable volume and is normally a rigid body, instead of a particle. For example, a prefabricated steel beam has a non-negligible volume and should be regarded as a rigid body in trajectory planning. As the example in Fig. 4 illustrates, a robotic manipulator hoists horizontally a purlin and moves forward along its linear track to assemble the purlin into a steel frame. If the trajectory planning is performed considering the end-tip of the robotic platform as a particle setting, the manipulated purlin might collide with the column of the steel frame. With this technical question for the robotic assembly of COVID-19 healthcare facilities, the authors further reviewed the existing studies in the robotics field (as enumerated in section 2). However, it is found that there is a lack of such rigid body-based solution in the robotics field.

In our approach, the trajectories of each building component are checked for collision against all the other components in an assembly environment. This research aims to develop a refined variant of the RRT* algorithm to realise the detour of a planned trajectory based on the manipulated object’s rigid body geometry.

4. System architecture of the approach

There are three levels in the hierarchy of the developed approach, namely assembly-level planning, manipulator-level planning, and joint-level planning. The levels are introduced in greater detail in the following subsections.

4.1. Assembly-level planning

Revit hosts an informative database for its projects, where the attributes of prefabricated components are categorised and structurally stored in the database in the form of identifiers (e.g., element ID, element category, coordinate in the BIM reference system). The example for a pillar element is presented in Fig. 5. As can be seen, the pillar (globally unique identifier: 657619) is registered in the Revit database with a property set of 17 identifiers. Specifically, Identifier #1 denotes the centreline which the purlin is drawn upon in Revit. Identifier #2 specifies where the base of the purlin is constrained to (i.e., the floor level on which the base of the purlin is placed). Identifier #3 marks the distance by which the base of the purlin is positioned above/below the corresponding floor level. Identifier #4 is where the top edge of the purlin is constrained to (i.e., the floor level on which the top edge of the purlin is placed). Identifier #5 denotes the distance by which the top edge of the purlin is positioned above/below the corresponding floor level. Identifier #6 implies if the purlin element is a structural component. Identifier #7 specifies the structural usage of the purlin element (i.e., horizontal bearings). Identifier #8 keeps record of the shape of the purlin. Identifiers #9, #10, #11, #12, #13, #14 illustrate the dimensions of the purlin, including length, width, height, perimeter, thickness, and volume. The material of the purlin is marked and codified by Identifiers #15 and #16 respectively. Identifier #17 specifies the coordinate of the purlin in the Revit model’s reference system.

In assembly-level planning, an ordered list of assembly tasks is created, which specifies the assembly coordinates and sequence of
prefabricated building components. This is enabled by performing semantic analyses on the BIM model of a standard flatpack house. The basic idea of semantic analysis is to semantically query the aimed identifiers in the dataset to retrieve the corresponding attributes of prefabricated components [38]. Dynamo is the medium that democratizes the database of Revit, where users can code in Python to derive semantic analyses on the BIM model of a standard flatpack house. The process first generates a list of building components arranged in ascending order along the z-axis (from the bottommost to the topmost). Then, for components that indicate the same size of z-coordinate values, the ascending y-axis procedure takes place to sort the components along the y-axis (from the leftmost to the rightmost). If there are components with the same y-coordinate value, the ascending x-axis procedure starts to sort the components from the rearmost to the foremost. Consequently, an ordered list of assembly tasks is generated, which is the aimed input for joint-level planning. This list goes to the joint-level planning stage where detailed trajectories for each task are then planned. Note that the assembly coordinates of building components in the assembly-level planning stage are determined according to the BIM model’s reference system. In the following joint-level planning stage, the robotic manipulator’s base coordinate system is marked as the world coordinate system. The assembly coordinates are transferred into the manipulator’s base coordinate system for trajectory planning.

4.2. Manipulator-level planning

At this level, the configuration of robotic manipulators is specified. The model of the manipulator demonstrated in this study is KUKA KR 120 R3100 (work range: 3095 mm, rated payload: 120 kg), which consists of seven joints (Fig. 6). Joint 1 is a prismatic joint, which operates utilising vacuum adsorption to hoist building components [40] (Fig. 6). The gripper is designed for heavy-duty applications and can withstand a weight of up to 68 kg, which is capable to handle the heaviest structural component of a flatpack house (i.e., long-edge beam, 55.7 kg, see Fig. 3).

The dimensions of the end-tip working range of a KUKA KR 120 R3100 manipulator is presented in Fig. 7a (i.e. the grey sphere in Fig. 7a). Note that the working range translates along with the Joint 1 linear displacement movement. However, it is found that the working range of a KUKA KR 120 R3100 manipulator cannot fully cover the spatial extent of a flatpack house for assembly, with the far end being out of reach (Fig. 7a). Given this, the collaborative assembly using paired KUKA KR 120 R3100 manipulators on either side of a flatpack house is considered (Fig. 7b).

4.3. Joint-level planning

In joint-level planning, the approach generates the joint motions for the robotic manipulator to accomplish various tasks. The flatpack house assembly process contains two states that need to be paid attention to for collision avoidance:

1) State #1: After a prefabricated component is hoisted from the material storage area (Fig. 8a), the joint 2 of the manipulator rotates in an anticlockwise direction to have its end-tip (A point in Fig. 8a) move from a storage area to an assembly workspace. To further transport the component to its assembly coordinate, the manipulator translates a linear displacement along its y-axis. As a result, the manipulated component could be in a state (A-B orientation in Fig. 8a) that runs into the front edge of the flatpack house. To avoid potential collisions, the trajectory plan aims to retract the end-tip of the manipulator from the distance \( d_{0A} \) in Fig. 8a to \( d_{A1} \) in Fig. 8b, where the end-tip A’ (after the retraction) is inside the collision-free

---

Fig. 3. Mass of prefabricated components and the number of each component used in a standard flatpack unit.

Fig. 4. Robotic assembly of a purlin into a steel frame.
aisle between the flatpack house and the linear track (see \( d_{02} \) in Fig. 8b). Then the end-tip undergoes a clockwise rotation to align the manipulated component along the \( y \)-axis, where the manipulator can translate along the \( y \)-axis and the potential collision with the front edge of the flatpack house is avoided. The manipulator configurations corresponding to the states in Fig. 8a and b are presented in Fig. 8c and d respectively.

2) State 2#: After the manipulator end-tip \( A \) aligns with the coordinate \( E \) on the \( y \)-axis (Fig. 8a), the assembly further approaches to the coordinate \( E \) along the \( x \)-axis. This makes the end-tip (as well as the manipulated building component) go entering the interior space of the flatpack house. Therefore, the manipulated component could be in a state (\( A-B \) orientation in Fig. 9a) that runs into the pillar of the flatpack house. To avoid potential collisions, the trajectory plan aims to have the end-tip undergo a clockwise rotation to align the manipulated component along the \( z \)-axis (Fig. 9b), where the
Fig. 7. Robotic platforms: (a) individual manipulator (vertical view, side view); (b) paired manipulators.
potential collision with the pillar is avoided. Then, the end-tip $A$ moves in the Oxy plane to assemble the component at the designated coordinate $E$. The collision avoidance problem becomes to plan the robot motion (the end-tip position and orientation) to constrain the geometry of the manipulated component in the obstruction-free area between the top and bottom frames (Fig. 9c).

To deal with the limitation of the RRT* algorithm for collision checks with rigid body constraints in states 1# and 2#, the algorithm is refined to incorporate the Minkowski Difference computation. The computation checks whether two shapes $A$ and $B$ have points in common by applying the symmetric difference function ($A \ominus B = \{a \ominus b | a \in A, b \in B\}$, where $a$ and $b$ represent random points in the geometric models of shapes $A$ and $B$ respectively) \cite{41}. If they do have a point in common, then $a \ominus b = 0$ exists in the $A \ominus B$ set and the collision is detected. The main body of the algorithm is presented in Algorithm 1. Given $\chi$ as the configuration space for the trajectory planning problem and $\chi_{obs}$ as the obstacle space, the collision-free space can be calculated as $\chi_{free} = \chi - \chi_{obs}$. The InitialiseTree function initialises an empty set $T$, which will be assigned with vertexes to form a trajectory for the end-tip of the manipulator. The InsertNode function adds the start state $x_{start}$ of each prefabricated building component $obj$ (to be assembled) as the root vertex of $T$. Note that the state $x_{start}$ marks both the position and orientation of the component (as well as the end-tip). The Sample function generates a random state $x_{rand}$ in $\chi_{free}$ based on a uniform distribution, which means that the probability of the potential position of $x_{rand}$ at any point within $\chi_{free}$ is equal. The Extend function is used to extend the trajectory tree $T$ growth preferentially toward the random state $x_{rand}$ (as generated by the Sample function) in the unexplored region of the collision-free space $\chi_{free}$. After the Nearest function finds the state $x_{nearest}$ in the trajectory tree $T$ that is closest to the random state $x_{rand}$ in the collision-free space $\chi_{free}$, the Extend function generates a new state $x_{new}$ along the direction from $x_{nearest}$ to $x_{rand}$ at a time step $\Delta t$ of the state progression of the robotic manipulator. The Near function collects a set of states $X_{near}$ in $\chi_{free}$ that locate in a spherical space with $x_{new}$ as the centre and $r$ as the radius. The ChooseMin function evaluates the time cost from $x_{start}$ to $x_{new}$ through each state in $X_{near}$ and identifies the state $x_{mid}$ with the lowest cost.

The IF function checks whether the manipulated building component $obj$ and the environmental objects share points in common along the edge between the $x_{new}$ and $x_{mid}$ states. If not, the manipulated building component $obj$ is in the collision-free space $\chi_{free}$. In particular, the points

Fig. 8. State #1 collision avoidance: (a) before retraction of the end-tip; (b) retraction of the end-tip and alignment of the manipulated building component; c) manipulator configuration before retraction of the end-tip; d) manipulator configuration after retraction of the end-tip and alignment of the manipulated building component.
in common between shapes are obtained by the functional modules Geometry Registrar, Indexed Faceset, and Minkowski Difference in the developed RRT* algorithm.

- First, the Geometry Registrar module registers, and pushes down, the geometric models of various objects in the robot construction environment to a stack trace, which sorts the registered geometric models into two collision detection parties—active and passive colliders. The active collider refers to the building component $obj$ being manipulated by the end-tip of the robotic manipulator, which is navigated to be assembled at the designated coordinate and “actively” looks for collisions with other environmental objects during the assembly process (see Fig. 10). The passive collider consists of the other environmental objects $\chi_{\text{obs}}$ such as the links of the robotic manipulator, terrain, and already-assembled-in-place building components, which “passively” await potential collisions to take place (see Fig. 10).

- Then, the Indexed Faceset module delineates the active and passive colliders as polyhedral formed by constructing faces (polygons), and

Fig. 9. State #2 collision avoidance: (a) before retraction of the end-tip; (b) rotation of the end-tip and alignment of the manipulated building component; c) displacement of the end-tip in the $Oxy$ plane.
indexes the vertices and normal of each constructing face into coordinates in the world coordinate system. The purple lines in Fig. 11 indexes the normal of each constructing face. By doing this, the algebraic representations of the active and passive colliders are developed (Fig. 11).

- Finally, the Minkowski Difference module checks whether the active and passive colliders have points in common by applying the symmetric difference function. This function works on the algebraic representations of the active and passive colliders to look for common points \( \text{obj} \ominus \chi_{\text{obs}} = \{a - b | a \in \text{obj}, b \in \chi_{\text{obs}}\} \), where \( a \) and \( b \) represent random points in the algebraic representations of the active and passive colliders \( \text{obj} \) and \( \chi_{\text{obs}} \) respectively. If the Minkowski Difference \( \text{obj} \ominus \chi_{\text{obs}} \) result returns zero, the common points between the active and passive colliders exist, which indicates that the collision is true.
If the IF condition is met, the Parent function takes $x_{new}$ as the parent node of $x_{mid}$, and the Line function adds the $x_{new} \leftarrow x_{mid}$ edge to the trajectory tree $T$. In addition, the Kinodynamic Rewiring function is used to smooth the edges in the generated trajectory tree $T$. The edges are the straight lines between the neighbouring states in a trajectory, which would imply sharp turns and cause the robot to perform sudden stops and change orientations right after. This could result in abrupt changes in robot's velocity from driving to steering. After the Line function adds the $x_{new} \leftarrow x_{mid}$ edge to the trajectory tree $T$, the RRT* algorithm integrates the Kinodynamic Rewiring function to smoothen the edges. The function interpolates feature points in the collision-free space $X_{free}$ as close as possible to the edges, and asymptotically adapt the edges by constructing a parametric curve to pass through all the feature points and neighbouring states in the trajectory tree $T$. Note that the developed RRT* algorithm has iterative innerworkings. Once the assembly trajectory for a building component is established, the geometry of the component (at the goal position) $obj$. Geometry. $x_{goal}$ is added to $X_{obs}$ for the trajectory planning in respect to the subsequent building components. After a trajectory is determined, the InverseKinematics function calculates the joint motions $[\text{joint}_1(t), \text{joint}_2(t), \ldots, \text{joint}_7(t)]$ that fulfil the kinematic constraints of each joint for the robotic manipulator to achieve the desired end-effector positions in time series $T(t)$.

**Algorithm 1.** Refined RRT*

```
1: T ← InitialiseTree();
2: T ← InsertNode($x_{start}$);
3: $j$ ← Ø;
4: $t$ = 0;
5: for $i = 1 \rightarrow N$ do
6:     $x_{rand}$ ← Sample($X_{free}$);
7:     $x_{nearest}$ ← Nearest($T$, $x_{rand}$);
8:     $x_{new}$ ← Extend($x_{nearest}$, $x_{rand}$, $\Delta t$);
9:     $x_{near}$ ← Near($T$, $x_{new}$);
10:    $x_{mid}$ ← ChooseMin($x_{new}$, $x_{near}$, $x_{start}$);
11:    if $0 \notin obj \cap X_{obs} = \{a - b | a \in obj, b \in X_{obs}, x_{new} \leftarrow x_{mid}\}$ then
12:        $T$ ← InsertNode($x_{new}$, $x_{mid}$; $T$);
13:        Parent($x_{new}$, $x_{mid}$);
14:        $T$ ← Line($x_{new}$, $x_{mid}$);
15:        Kinodynamic Rewiring($T$, $x_{new}$, $x_{mid}$);
16:    end if
17:    if $\|x_{new} - x_{goal}\| < d$ then
18:        $X_{obs}$ ← obj. Geometry. $x_{goal}$;
19:        Break;
20:    end if
21: end for
22: return $T$;
23: do
24:    $[\text{joint}_1(t), \text{joint}_2(t), \ldots, \text{joint}_7(t)]$ ← $T(t)$. InverseKinematics;
25:    $f$ ← $[\text{joint}_1(t), \text{joint}_2(t), \ldots, \text{joint}_7(t)]$;
26:    $t$ = $t + \Delta t$;
27:    while $\|T(t) - x_{goal}\| > \epsilon$ do
28: end do − while
29: return $f$;
```

Although the locations for assembling each building component are predetermined in the assembly-level planning, the authors acknowledge that the obstacles in the operational environment are well-known and obvious to the observers of the platform rather than the robot itself. This is because the obstacles in the environment are gradually accumulated during the assembly process. For example, a previously assembled column could become the obstacle in the assembly of a subsequent component (e.g., purlin) (Fig. 12a). Therefore, the obstacle space boundary in the robot operating environment is subject to change from
time to time, which is dynamic and unpredictable for the robot. In this regard, algorithm 1 is carried out online and has inner-workings to 1) iteratively check and register geometric models of newly introduced obstacles in real-time (Fig. 12b), 2) generate robots’ kinematic parameters for realising the assembly actions in a collision-free manner, and 3) publish the generated kinematic data to each joint of the robots for instructing real-time operations. In addition, the material storage location is fixed and regarded as the assembly start state of each building component in Algorithm 1 for trajectory planning.

5. Testing of the approach

The planning efficiency of the developed approach was tested in three perspectives: 1) collision avoidance, 2) trajectory smoothness, and 3) trajectory length and execution time. The results were presented in the following subsections.

5.1. Collision avoidance

The developed approach was tested in a simulated construction environment. During the assembly of the flatpack building components, the approach exhibited satisfactory collision avoidance performance. Fig. 13 presents an example of the trajectory planned for purlin assembly. The trajectory as illustrated sequentially in the sub-figures is interpreted in the figure caption.

To quantitatively evaluate the obstacle avoidance performance of the developed approach, the Minkowski distances between the obstacle space $X_{obs}$ and the manipulated building objects $obj$ at different states along the planned trajectories were calculated. The Minkowski distance is a minimum distance measurement between 3D geometries [41]. Taking the different states in the purlin assembly process illustrated in Fig. 13 as an example. The Minkowski distances are calculated and the results are presented in Table 1. As can be seen, the Minkowski distances at different states are greater than or equal to zero, which indicates that the manipulated purlin at different states along the planned trajectory is kept in a distance from the obstacle space $X_{obs}$ and the collision is avoided. Note that the distances calculated for the states 9 and 10 are zero, which means that the purlin is assembled into the steel frame and is thereby in contact with the obstacle space $X_{obs}$.

5.2. Trajectory smoothness

Applying the sampling-based method, the trajectory might be jerky given that the planning is performed by connecting states in time series and there could be sharp turns. As introduced in section 4.3, the Kinodynamic Rewiring function is useful for trajectory smoothing and was utilised in algorithm 1. The function asymptotically optimised sharp turns in the trajectory planning in this research and resulted in smooth end-tip displacements. Taking the purlin assembly in Fig. 13 as an example, not only the position trajectory but also the orientation trajectory is smooth (see Fig. 14).

To quantitatively evaluate the trajectory smoothing performance of the developed approach, the derivatives $\dot{x}(t), \dot{y}(t), \dot{z}(t)$ of the planned trajectories were examined. The criteria for a smooth trajectory are that...
Fig. 13. An example of the trajectory planned for purlin assembly: (a) and (b) show that the robotic manipulator hoists the purlin from the material storage area and rotates joint 2 to get close to the assembly workspace; (c) shows that joints 3 and 4 of the manipulator adjust their respective angles cooperatively to keep the end-tip (as well as the purlin) in a distance from the already-in-place flatpack building components (i.e. the pillar); (d) shows that the manipulator moves forward along joint 1 axis to get close to the assembly coordinate; (e), (f), (g), and (h) show that joints 3, 4, 5, 6, and 7 of the manipulator adjust their respective angles cooperatively to send the purlin into the interior of the flatpack unit in the meantime avoid the purlin-obstacle collision; (i) shows that joints 4, 5, 6, and 7 of the manipulator adjust their respective angles cooperatively to place the purlin at the designated assembly coordinate; and (j) shows that the manipulator returns to the initial configuration after performing the assembly task.
its derivatives $x(t)$, $y(t)$, $z(t)$ are continuous and not simultaneously zero [19]. Taking the purlin assembly trajectory in Fig. 14 as an example. The derivatives $x(t)$, $y(t)$, $z(t)$ are computed and the results are plotted in Fig. 15. As can be seen, the derivatives $x(t)$, $y(t)$, $z(t)$ are continuous and not simultaneously zero, which indicates that the trajectory is smooth.

5.3. Trajectory length and execution time

The results on the average trajectory length and execution time of prefabricated components in each category, and the total execution time are presented in Table 2 below. It was recorded that the total execution time was approximately 28 min (two robotic manipulators working simultaneously) (see Table 2). The authors consulted industrial professionals from contractors and learned that the expected assembly time on-site for two persons working simultaneously is approximately 45 min. The construction time seems shorter utilising robotic manipulators. Note that this research aimed at planning collision-free trajectories for robotic manipulators to place the prefabricated components at the designated location. There is a further auxiliary procedure involved, which is to bolt the prefabricated components. The total execution time could be inflated to account for the bolting time.

6. Discussion

This research developed a collision-free trajectory planning approach that can assist in the robotic assembly of COVID-19 healthcare facilities. In the existing literature in the construction field [9–11,42–44], the authors found that the robot end-tips (as well as the manipulated building components) are represented in the particle setting for trajectory planning. However, the prefabricated building components of COVID-19 healthcare facilities have nonnegligible volume, which might be better described as rigid bodies rather than particles. If the trajectory planning is performed considering the end-tip of the robotic platform (as well as the manipulated building component) as a particle setting, the robot may collide with surrounding objects. The developed approach fills in the identified knowledge gap by refining the RRT* algorithm to incorporate geometry-based collision checks. The collision detection further enables the detour of a planned trajectory based on the geometry of the prefabricated building components of COVID-19 healthcare facilities.

The planning efficiency of the developed approach was tested in three perspectives: 1) collision avoidance, 2) trajectory smoothness, and 3) trajectory length and execution time. To quantitatively evaluate the obstacle avoidance performance of the developed approach, the Min-kowski distances between the obstacle space $J_{obb}$ and the manipulated building objects $obj$ at different states along the planned trajectories were calculated. The results indicated that the manipulated building components were kept in a distance from the obstacle space $J_{obb}$ during their assemblies and the potential collisions were avoided. To quantitatively evaluate the trajectory smoothing performance of the developed approach, the derivatives $x(t)$, $y(t)$, $z(t)$ of the planned trajectories were examined. The authors found that the derivatives $x(t)$, $y(t)$, $z(t)$ were continuous and not simultaneously zero, which proves that the planned trajectories are smooth. In addition, the average trajectory length and execution time of prefabricated components in each category were measured, and the results are as follows: beam—short edge (2.780 m, 23 s), beam—long edge (6.781 m, 58 s), purlin (4.309 m, 34 s), floor panel (3.504 m, 28 s), pillar (4.692 m, 39 s), roof panel (6.054 m, 49 s), and wall panel (3.804 m, 30 s). Compared with human workers involved construction activities, the results seem labour- and time-saving.

Statistics suggest that there are regions experiencing COVID-19 patient surge while having insufficient healthcare capacity for patient care [5]. However, human activities increase COVID-19 transmission. To prevent the spread of the virus, technologies that can enable the smooth delivery of hospitalisation facilities under pandemic circumstances are therefore in urgent need. Robots do not get the biological virus and thus ensure that the essential construction tasks are not suspended during the pandemic. The developed approach could be labour- and time-saving. It was recorded that the total assembly time was approximately 28 min applying the approach (see section 5.3), whereas the time on site for two persons working simultaneously seems to be longer—45 min (suggested by industry professions). Note that the total robotic assembly time could be inflated to account for a further auxiliary procedure, which is to bolt the prefabricated components. This research aimed at planning collision-free trajectories for robotic manipulators to place the prefabricated components at the designated location. In the subsequent research, the authors will investigate further the use of a collaborative robot (e.g., aerial operation robot) to assist in screwing bolt connections after the components are placed at the designated locations.

7. Conclusion and future work

This research presents a collision-free trajectory planning approach for robotic assembly of lightweight structures for COVID-19 healthcare facilities with prefabricated components. The approach is developed into three levels, namely assembly-planning level, manipulator-planning level, and joint-planning level. The three levels function in a successive sequence to generate plans for the robotic manipulators to accomplish assembly tasks and avoid potential collisions. In particular, the assembly-planning level creates an ordered list of assembly tasks, which consists of the assembly coordinates and sequence of the prefabricated building components that compose the COVID-19 healthcare facilities. The assembly coordinates represent the end-tip positions of the robotic manipulators in the trajectory planning. The manipulator-planning level determines the configuration of the robotic manipulator utilised to perform the assembly tasks. At last, the joint-planning level receives information from the former levels, and calculates the joint motions that fulfil the kinematic constraints of each joint for the robotic manipulators to achieve the desired end-effector positions in time series and assemble the components into designated coordinates. Given that the prefabricated building components of COVID-19 healthcare facilities have non-negligible volume, the approach is developed to incorporate

| States | Translations (m) | Orientations (rad) | Minkowski Distances (m) |
|--------|------------------|--------------------|-------------------------|
| 1 (Fig. 13a) | (0, −2.50, 1.00) | (3.14, 0, −1.57) | 1.21 |
| 2 (Fig. 13b) | (2.37, 0.32, 1.03) | (3.14, 0, 3.14) | 0.12 |
| 3 (Fig. 13c) | (1.75, 1.50, 2.46) | (3.14, 0, 3.14) | 0.56 |
| 4 (Fig. 13d) | (1.75, 3.20, 2.46) | (3.14, 0, 3.14) | 0.56 |
| 5 (Fig. 13e) | (1.76, 3.20, 2.97) | (0, −1.57, 0) | 0.54 |
| 6 (Fig. 13f) | (1.55, 4.50, 2.57) | (0, −1.57, −0.52) | 0.17 |
| 7 (Fig. 13g) | (1.28, 5.50, 2.02) | (0, −1.57, −1.57) | 1.73 |
| 8 (Fig. 13h) | (1.88, 5.50, 2.48) | (0, −1.57, −1.57) | 0.54 |
| 9 (Fig. 13i) | (2.82, 7.00, 2.95) | (0, −1.57, −1.57) | 0.00 |
| 10 (Fig. 13j) | (0,0,0) | (0, 0, 0) | 0.00 |
geometry-based collision checks by utilising the Minkowski Difference computation. This enables the detour of a planned trajectory based on the geometry of a prefabricated building components to prevent an impending collision. This research thereby fills in the knowledge gap in the construction field literature by providing an approach that generates trajectory plans for prefabricated building components with non-negligible volumes.

The planning efficiency of the developed approach was evaluated in three perspectives: 1) collision avoidance, 2) trajectory smoothness, and 3) trajectory length and execution time. The results revealed that the approach has satisfactory performance in these three criteria. However, this research has the following limitations and subsequent research needs to be conducted. This research focused on the development of an approach to plan collision-free trajectories for robotic manipulators to place the prefabricated components at the designated location. There is a further auxiliary procedure involved in order to realise a completely automatic processing in robotic assembly of COVID-19 healthcare facilities—flatpack house, which is to bolt the prefabricated

![Fig. 14. Purlin assembly trajectory: (a) front view; (b) back view.](image-url)
components. In the subsequent research, the authors will investigate further the use of a collaborative robot (e.g., aerial operation robot) to assist in screwing bolt connections after the components are placed at the designated locations.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgement

This work was partially supported by National Key R&D Program of China under Grant No. (2018YFE0125400), China’s National Natural Science Foundation under Grant No. (52108179), and Science and Technology Research Project of Zhejiang Province Transportation Department under Grant No. (202217).

References

[1] M.B. Esa, F.S.B. Ibrahim, E.B.M. Kamal, Covid-19 pandemic lockdown: the consequences towards project success in Malaysian construction industry, Adv. Sci. Technol. Eng. Syst. J. 5 (2020) 973–983, https://doi.org/10.25046/05050119.
[2] K. Simonson, Construction Data, the Associated General Contractors of America. https://www.agc.org/learn/construction-data, 2021.
[3] Construction Industry High Quality Development Research Institute, Investigation report on how Covid-19 impacted the Chinese construction enterprises. http://www.zgyjzy.org.cn/menu20/newsDetail/8428.html, 2020 (accessed August 21, 2021).
[4] W. Wemhun, F. Yuting, G. Jia, S. Ke, G. Shuke, J. Jinwen, N. Guodong, Y. Zhenmin, Q. Yaning, M. Lingyun, How the COVID-19 outbreak affected organizational citizenship behavior in emergency construction megaprojects: case study from two emergency hospital projects in Wuhan, China, J. Manag. Eng. 37 (2021) 4021008, https://doi.org/10.1061/(ASCE)ME.1943-5479.0000922.
[5] S. Brakman, H. Garretsen, A. van Witteloostuijn, Robots do not get the coronavirus: the COVID-19 pandemic and the international division of labor, J. Int. Bus. Stud. 52 (2021) 1215–1244, https://doi.org/10.1111/jibs.12419.
[6] N. King, M. Bechthold, A. Kane, P. Michalatos, Robotic tile placement: tools, techniques and feasibility, Autom. Constr. 39 (2014) 161–166, https://doi.org/10.1016/j.autcon.2013.06.001.
[7] Y. Meng, Y. Sun, W. Chang, Optimal trajectory planning of complicated robotic timber joints based on particle swarm optimization and an adaptive genetic algorithm, construction, Robotics. 5 (2021) 131–146, https://doi.org/10.1007/s41693-021-00057-w.
[8] Y. Terada, S. Murata, Automatic modular assembly system and its distributed control, Int. J. Robot. Res. 27 (2008) 445–462, https://doi.org/10.1177/0278364907085562.
[9] O. Davtalab, A. Kazemian, B. Khoshnevis, Perspectives on a BIM-integrated software platform for robotic construction through contour crafting, Autom. Constr. 89 (2018) 13–23, https://doi.org/10.1016/j.autcon.2018.01.006.
[10] O. Kontouvakis, G. Tryfonos, C. Georgiou, Robotic additive manufacturing (RAM) with clay using topology optimization principles for nozzlepath planning: the example of a building element, Archit. Sci. Rev. 63 (2020) 105–118, https://doi.org/10.1003/00038628.2019.1620170.
[11] L. Ding, W. Jiang, Y. Zhou, C. Zhou, S. Liu, BIM-based task-level planning for robotic brick assembly through image-based 3D modeling, Adv. Eng. Inform. 43 (2020), 100993, https://doi.org/10.1016/j.aei.2019.100993.
[12] W. Xiao, C.G. Cassandras, C.A. Belata, Bridging the gap between optimal trajectory planning and safety-critical control with applications to autonomous vehicles, Automat. 129 (2021), 101992, https://doi.org/10.1016/j.automa.2021.109992.
[13] A. Vagale, R. Ouchekhi, R.T. Bye, O.L. Osen, T.I. Fossen, Path planning and collision avoidance for autonomous surface vehicles I: a review, J. Mar. Sci. Technol. 26 (2021) 1292–1306, https://doi.org/10.1007/s10877-020-00976-6.
[14] O. Sousini, B. Renzallat, D. Duvivier, A. Artiba, N. Belanger, P. Feyzeau, Path planning: A 2013 study, in: Proceedings of 2013 International Conference on Industrial Engineering and Systems Management (IEMS), 2013, pp. 1–8. https://ieeexplore.ieee.org/document/676152.
[15] D.M. Saxena, S. Bae, A. Nakhaei, K. Fujimura, M. Likhachev, Driving in dense traffic with model-free reinforcement learning, in, IEEE Int. Conf. Robot. Automat. (ICRA) 2020 (2020) 5385–5392, https://doi.org/10.1109/ICRA4945.2020.9177152.
[16] L.E. Kavraki, P. Svestka, J.-J. Latombe, M.H. Overmars, Probabilistic roadmaps for path planning in high-dimensional configuration spaces, IEEE Trans. Robot. Autom. 12 (1996) 566–580, https://doi.org/10.1109/70.508459.
[17] J. Borenstein, Y. Koren, Real-time obstacle avoidance for fast mobile robots, IEEE Trans. Syst. Man Cybernet. 19 (1989) 1179–1187, https://doi.org/10.1109/21.440003.
[18] P. Bhattacharya, M.L. Gavrilova, Roadmap-based path planning using the Voronoi diagram for a clearance-based shortest path, IEEE Robot. Automat. Mag. 15 (2008) 58–66, https://doi.org/10.1109/MRA.2008.921540.
[19] X. Fan, Y. Guo, H. Liu, B. Wei, W. Lya, Improved artificial potential field method applied for UAV path planning, Math. Prob. Eng. 2020 (2020) 6523558, https://doi.org/10.1155/2020/6523558.
[20] P. Wang, S. Gao, L. Li, B. Sun, S. Cheng, Obstacle avoidance path planning design for autonomous driving vehicles based on an improved artificial potential field algorithm, Energies. 12 (2019) 2342, https://doi.org/10.20945/2424-9992.
[21] A. Ataka, H. Lam, K. Althofer, Reactive magnetic-field-inspired navigation for non-holonomic mobile robots in unknown environments, in: 2018 IEEE International Conference on Robotics and Automation (ICRA), 2018, pp. 6986–6988, https://doi.org/10.1109/ICRA.2018.846320.
[22] M. Becker, T. Litge, M.A. MILLER, S. Haddadin, Circular fields and predictive multi-agents for online global trajectory planning, IEEE Robot. Aut. Lett. 6 (2021) 2618–2625, https://doi.org/10.1109/LRA.2021.3061997.
[23] L. Petrovic, J. Peric, M. Seder, J. Markovic, Stochastic optimization for trajectory planning with Heteroscedastic Gaussian processes, in: 2019 European Conference on Mobile Robots (ECMR), 2019, pp. 1–6, https://doi.org/10.1109/ECMR.2019.8979970.
[24] R.A. Shaym, P. Lightbody, G. Das, P. Liu, S. Gomez-Gonzalez, G. Neumann, Improving local trajectory optimisation using probabilistic movement primitives, in: 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2019, pp. 2666–2671, https://doi.org/10.1109/IROS.2019.8879846.
[25] N. Ratliff, M. Zucker, J.A. Bagnell, S. Srinivasa, CHOMP: gradient optimization techniques for efficient motion planning, in: 2009 IEEE International Conference on Robotics and Automation, 2009, pp. 489–494, https://doi.org/10.1109/ROBOT.2009.5152517.
[26] A. Bryavan, B. Boots, S.S. Srinivasa, D. Fox, Space-time functional gradient optimization for motion planning, in: 2014 IEEE International Conference on Robotics and Automation (ICRA), 2014, pp. 6499–6506, https://doi.org/10.1109/ICRA.2014.6907981.
[27] M. Zucker, N. Ratliff, A.D. Dragan, M. Pivtorakio, M. Klingensmith, C.M. Dellin, J. A. Bagnell, S.S. Srinivasa, CHOMP: covariant Hamiltonian optimization for motion planning, Int. J. Robot. Res. 32 (2013) 1164–1193, https://doi.org/10.1177/0278364912468080.
[28] K. He, E. Martin, M. Zucker, Multigrad CHOMP with local smoothing, in: 2013 13th IEEE-RAS International Conference on Humanoid Robots (Humanoids), 2013, pp. 315–322, https://doi.org/10.1109/HUMANOIDS.2013.709995.
[29] M. Kalakrishnan, S. Chitta, E. Theodorou, P. Pastor, S. Schaal, STOMP: stochastic trajectory optimization for motion planning, in: 2011 IEEE International Conference on Robotics and Automation, 2011.
Conference on Robotics and Automation, 2011, pp. 4569–4574, https://doi.org/10.1109/ICRA.2011.5980280.

[30] J. Schulman, Y. Duan, J. Ho, A. Lee, I. Awwal, H. Bradlow, J. Pan, S. Patil, K. Goldberg, P. Abbeel, Motion planning with sequential convex optimization and convex collision checking, Int. J. Robot. Res. 33 (2014) 1251–1270, https://doi.org/10.1177/0278364914528132.

[31] J. Cortés, T. Siméon, in: M.H. Ang, O. Khatib, B. Siciliano (Eds.), Sampling-Based Tree Planners (RRT, EST, and Variations) BT - Encyclopedia of Robotics, Springer, Berlin Heidelberg, Berlin, Heidelberg, 2020, pp. 1–9, https://doi.org/10.1007/978-3-642-41610-1_170-1.

[32] S.M. LaValle, J.J. Kuffner, Randomized kinodynamic planning, Int. J. Robot. Res. 20 (2001) 378–400, https://doi.org/10.1177/027836490122067453.

[33] J.D. Gammell, S.S. Srinivasan, T.D. Barfoot, Informed RRT*: Optimal sampling-based path planning focused via direct sampling of an admissible ellipsoidal heuristic, in: 2014 IEEE/RSJ International Conference on Intelligent Robots and Systems, 2014, pp. 2997–3004, https://doi.org/10.1109/IROS.2014.6942976.

[34] V. Bushell, L. Thomas, J. Combes, Inside the O2: the NHS nightingale hospital, J. Interprof. Care. 34 (2020) 698–701, https://doi.org/10.1080/13561820.2020.1823940.

[35] T. Hale, N. Angrist, R. Goldszmidt, B. Kira, A. Petherick, T. Phillips, S. Webster, E. Cameron-Blake, L. Hallas, S. Majumdar, H. Tatlow, A global panel database of pandemic policies (Oxford COVID-19 government response tracker), Nat. Hum. Behav. 5 (2021) 529–538, https://doi.org/10.1038/s41562-021-01079-8.

[36] R.C. Khanna, M.V. Cicinelli, S.S. Gilbert, G.S.V. Murphy, COVID-19 pandemic: lessons learned and future directions, Indian J. Ophthalmol. 68 (2020) 703–710, https://doi.org/10.4103/ijio.IJO_843_20.

[37] H. Luo, J. Liu, C. Li, K. Chen, M. Zhang, Ultra-rapid delivery of specialty field hospitals to combat COVID-19: lessons learned from the Leishenshan hospital project in Wuhan, Autom. Constr. 119 (2020), 103345, https://doi.org/10.1016/j.autcon.2020.103345.

[38] F. Tang, T. Ma, J. Zhang, Y. Guan, L. Chen, Integrating three-dimensional road design and pavement structure analysis based on BIM, Autom. Constr. 113 (2020), 103152, https://doi.org/10.1016/j.autcon.2020.103152.

[39] KUKA, KUKA KR 120 R3100. https://www.kuka.com/en-cn/products/robotics-systems/industrial-robots/kr-quantec, 2021 (accessed August 21, 2021).

[40] M. Huang, L. He, D. Choi, J. Pecchia, Y. Li, Picking dynamic analysis for robotic harvesting of Agaricus bisporus mushrooms, Comput. Electron. Agric. 185 (2021), 106145, https://doi.org/10.1016/j.compag.2021.106145.

[41] A. Yang, Q. Liu, W. Naeem, M. Fei, Robot dynamic collision detection method based on obstacle point cloud envelope model, in: Q. Han, S. McLough, C. Peng, B. Zhang (Eds.), Intelligent Equipment, Robots, and Vehicles - 7th International Conference on Life System Modeling and Simulation, Springer Singapore, Singapore, 2021, pp. 370–378, https://doi.org/10.1007/978-981-16-7213-2_36.

[42] Y. Guo, J. Meng, J. Shu, Y. Liu, BIM-based task and motion planning prototype for robotic assembly of COVID-19 hospitalisation light weight structures, Autom. Constr. 140 (2022), 104370, https://doi.org/10.1016/j.autcon.2022.104370.

[43] W. Zhao, Y. Jiang, Y. Liu, J. Shu, Automated recognition and measurement based on three-dimensional point clouds to connect precast concrete components, Autom. Constr. 133 (2022), 104000, https://doi.org/10.1016/j.autcon.2021.104000.

[44] W. Zhao, Y. Liu, J. Zhang, Y. Shao, J. Shu, Automatic pixel-level crack detection and evaluation of concrete structures using deep learning, Struct. Control Health Monit. 29 (8) (2022), e2981, https://doi.org/10.1002/stc.2981.