An Extension of BIM Using AI: A Multi Working-Machines Pathfinding Solution

YUSHENG XIANG[1,2,3], **(Member, IEEE), KAILUN LIU**[2], **(Student Member, IEEE),**
TIANQING SU[1,4]**, (Member, IEEE), JUN LI**[1,5]**, (Member, IEEE), SHIRUI OUYANG**[2]**,**
SAMUEL S. MAO[1,3]**, AND MARCUS GEIMER**[2]

[1]Elephant Tech LLC, Shenzhen 518000, China
[2]Institute of Mobile Machines, Karlsruhe Institute of Technology, 76131 Karlsruhe, Germany
[3]Department of Mechanical Engineering, University of California at Berkeley, Berkeley, CA 94720, USA
[4]Guanghua School of Management, Peking University, Beijing 100081, China
[5]Center of AI, Faculty of Engineering and Information Technology, University of Technology Sydney, Ultimo, NSW 2007, Australia

Corresponding author: Yusheng Xiang (yusheng.xiang@partner.kit.edu)

ABSTRACT Multi working-machines pathfinding solution enables more mobile machines simultaneously to work inside of a working site so that the productivity can be expected to increase evolutionary. To date, the potential cooperation conflicts among construction machinery limit the amount of construction machinery investment in a concrete working site. To solve the cooperation problem, civil engineers optimize the working site from a logistic perspective while computer scientists improve pathfinding algorithms' performance on the given benchmark maps. In the practical implementation of a construction site, it is sensible to solve the problem with a hybrid solution; therefore, in our study, we proposed an algorithm based on a cutting-edge multi-pathfinding algorithm to enable the massive number of machines cooperation and offer the advice to modify the unreasonable part of the working site in the meantime. Using the logistic information from BIM, such as unloading and loading point, we added a pathfinding solution for multi machines to improve the whole construction fleet's productivity. In the previous study, the experiments were limited to no more than ten participants, and the computational time to gather the solution was not given; thus, we publish our pseudo-code, our tested map, and benchmark our results. Our algorithm's most extensive feature is that it can quickly replan the path to overcome the emergency on a construction site.

INDEX TERMS Smart working site, multi agents pathfinding, conflict based searching, building information model, construction site productivity, Huoshenshan hospital project, path planning.

I. INTRODUCTION

Although many achievements in construction machines with respect to productivity and safety, humans' pursuit of even higher productivity and better safety never stops. In the past decades, civil engineers and construction-industry-related software engineers introduce the Building Information Model (BIM) [1] as a powerful tool to increase productivity and safety performance whilst reduce the project cost by means of digital technology. In general, BIM provides the 3D or more than 3D model of the construction projects and even the installation sequence of the components to avoid mistakes during the construction stage. With the maturity of BIM, this software and process are adopted for

The associate editor coordinating the review of this manuscript and approving it for publication was Sun-Yuan Hsieh.

many large and especially complicated construction projects worldwide [2] since the mistakes during the real construction process cause much more severe consequences than those in virtual engineering. Also, BIM is considered as a life-long software, contributing to not only the early construction phase [1], [3] but also the time after the construction projects are finished [4]. Despite to the preliminary cost for training corresponding laborers and model building in a computer, BIM is a necessary tool for at least large construction projects becomes a consensus. However, although current BIM software defines the start and end points where the material should be transported, concrete paths guiding the trucks to accomplish the goal are not given. Or more generally, an algorithm that determines the paths of the participants in the working site so that they can move to their destination without collision and hesitation is still developing.
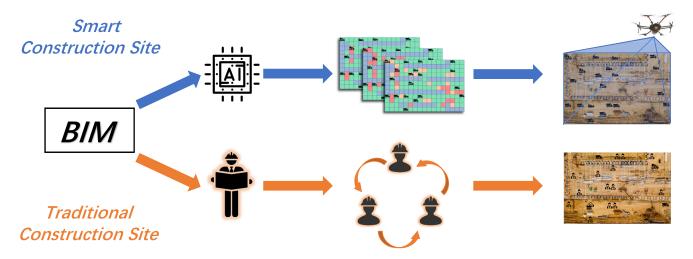
FIGURE 1. Overview of the smart construction site concept. We introduce to use artificial intelligence to unify the scheduling of construction machinery on construction sites. Maximize the use of construction machinery on specific construction sites by avoiding conflicts among construction machinery and thus ultimately improve the productivity of the construction sites.

As shown in Fig. 1, one motivation to combine these path planning algorithms with BIM can be described as determining the construction machines' travel path so that the machines can be expected to move faster and denser without hesitation. To accomplish higher productivity and better safety in the path planning fashion, some researchers calculate and display efficient paths by a given construction site [5], [6] whereas others optimize the construction site layout considering the high productivity of the path [7], [8]. However, most of the current solutions about path planning on a construction site are mainly focusing on individual units, i.e., the interaction and path conflict of the different machines or other agents are ignored. Consequently, the working sites' spatial and time utilization is limited. Inspired by warehouse logistics solution [9], where a lot of robotics are working with the commands from path planning algorithms in the meantime and thus achieve considerably higher transport efficiency as a fleet, we envision the multi-agents pathfinding (MAPF) solution can also provide evolutionary to the construction industry. A persuasive instance to show the benefit of introducing MAPF solution into a working site is the construction site for the famous hospital, namely Huoshenshan, in Wuhan during the coronavirus 2019 outbreak. Invested an extraordinary amount of working machines and human cooperators, the construction project was finished at an unprecedented speed, through only manually coordinate to avoid conflicts among machines. Apparently, the economic cost of running such a construction site can be quite expensive due to experienced workers. Also, since the MAPF problem is NP-hard, computer algorithms can surely have a better performance concerning a series of optimization objects, such as shorter moving distance and realtime performance. In light of that, we propose a MAPF solution for working machines as an extension of the BIM system so that more machines can work simultaneously and thereby achieve better holistic productivity.

The aim of this paper is to extend the current BIM software with Artificial Intelligence (AI) path-planning algorithms in order that more machines can work simultaneously since the paths are calculated to avoid collisions. Because the start points and endpoints can be given directly in the BIM system, we devote ourselves to the implementation level, i.e., how exactly the machines should achieve their goal set given by the BIM system. We envision that the case in Wuhan will be a normal case in the future by utilizing AI and IoT [10], [11] technology.

The main contributions of this paper can be sum up as the following points:

- We introduced a novel MAPF solution to guide a fleet of mobile machines to work simultaneously inside a working site and to give suggestions to optimize the construction site layout.
- The approach we proposed can always find a feasible solution on weighted maps considering the agents' priority in a predefined short period to deal with emergencies.
- We extended the cutting-edge MAPF solution with a bidirectional searching method for the initial search of the best path, which shall be principally faster.
- Our method can be added to BIM software to make up for its lack of path planning.
- We benchmark the performance of our MAPF solution for the mobile machines with respect to the solution found time and cost to reach their goal on the given maps.

The rest of this paper is organized as follows. Section II briefly introduces the prerequisite and background knowledge in fields of BIM, path planning methods for construction machines and robotics to understand this paper quickly. Next, the existing problems are illustrated in Section III. After that, in Section IV, we describe the setup of our MAPF approach, including the map to give the start and goal positions, low

level for individual pathfinding, and high level for conflicts solving. Then, we show the experiments setup. Followed by section VI, we show our approach's performance by testing on some maps based on real construction sites. Finally, Section VII summarizes the advantages of our approach, and Section VIII gives conclusions and envisions the outlook.

II. RELATED WORKS
A. BUILDING INFORMATION MODELLING
BIM is a 3D model-based information management process in the field of Architecture, Engineering, and Construction (AEC) that facilitates efficient design and construction processes and inter-organizational collaboration [12]. There is a lot of BIM-based software: Autodesk Revit Building (Revit), ArchiCAD, Bentley, and SolidWorks [13]. By building the whole project virtually before physical construction begins, construction sequencing is determined, including material ordering, fabrication, and delivery schedules for all building components, etc. Therefore, conflict, interference, and collision are avoided in the early stage, contributing to improved site efficiency and reduced cost [14]. As the function inside BIM increases, such as scheduling, virtual reality [15], and logistic management [12], it has been extended to 4 or more than 4 dimensional model. Nowadays, the research about BIM is prosperous. Combining AI and IoT into the BIM system is considered the next potential boom for BIM systems. Survey papers about that can be found in [16], [17].

In order to automate the whole construction site and compute the optimal path for the heavy machines, logistic information is vital, which demonstrates which materials should be placed in which location at which time in the right quantity. Logistics management in construction involves the strategic storage, handling, transportation and distribution of resources, as well as planning of a building site's layout [18]. Whitlock has proposed a desktop approach to adopt BIM for construction logistics management [12]. Such logistic information, for instance, unloading points, on-site arrangements-logistics layouts, which are generated at the outset of the pre-construction process, can be used as the input data for the path planning.

B. PATH PLANNING FOR CONSTRUCTION MACHINES
In a construction site, there are usually multiple machines working simultaneously in a definite area with given assignments. Therefore, coordinated construction logistics can definitely increase productivity, decrease material usage, and guarantee workers' health and safety [19]. To date, there is plenty of researchers proposed their solution for the logistical problem inside construction sites at diverse levels, i.e., path planning and motion control. In this section, we summarize the previous research about moving paths inside of construction sites.

In the construction industry, the earth-moving sector is among the pioneers in adopting new sensing and information technologies [20], such as bulldozer [21], [22], and grading machines [23]. Given two points A and B on a construction site, the objective is to determine the shortest path from A to B maintaining a safe distance from obstacles. The approach proposed by Kim is a path-planning method for a mobile construction robot to find a continuous collision-free path from the initial position of the construction robot to its target position by improved Bug-based algorithm [24]. The algorithm can work with the disturbance of static and dynamic objects. Obviously, the performance of the approach is based on the accuracy of the sensors. At that time, the methods to acquire site information were still immature. Hence, the spatial model supporting path planning in a partially known and partially unknown environment was brought forward by Lee. Accordingly, the spatial model provides the domain for finding an efficient path in a construction site through the use of an algorithm that combines a shortest path algorithm and a dynamic path-planning algorithm. This approach differs from existing path-planning approaches that assume the construction site is totally known or totally unknown. Thus, problems associated with managing a changing construction environment and ignorance of designated roadway networks are overcome [25]. In the same year, Soltani has compared the performance of different methods for the path planning inside construction sites [26], such as Dijkstra [27], A^* [28], and Genetic Algorithm (GA) [29], by evaluating comprehensive multi objects, e.g., site layout representation, distance formulation, hazard zone modeling, and visibility calculations. Also, the author points out the use of CCTV cameras should be considered to enhance site security [26]. Although the simulation results show that the GA has the best performance and the other two algorithms have quite similar results, we conjecture that it might be ascribed to their maps, which are quite easy and do not include difficult obstacles such as bottlenecks. Fairly recently, Song tried to integrate some path planning algorithms into BIM system [5]. The basic idea is to determine the path to transport the materials at the very beginning phase, i.e., the construction site design phase. Also, the study verified the demand for the introduction of path planning by survey questions. The shortcoming of this approach is that the interaction of other participants inside construction sites during the construction project was not taken into account. So far, the aforementioned studies focus on a path for one machine inside the construction site. In contrast, the following research shows solutions for multi machines working simultaneously on a construction site. An influential study about the path planning on the construction site is from Cheng published in 2012 [30]. The objective of the paper is to provide the n best and safest paths between two points in a work area while maintaining a safe distance from identified obstacles. Here the approach proposed in the paper uses Dijkstra algorithm and solves the path of different participants in sequence. Due to the limited recognition distance of ultra-wideband sensors, the usage of this approach might not suitable for huge working sites. Also, since the paths of agents are calculated one by one, the computation efficiency is not ideal from today's point

of view in 2020. 4 years after Cheng's study, the research from Bohacs shows the difficulties of path planning for a construction site. Also, they use A^* as basic and develop an algorithm to let limited machines can cooperate without collision [31] within a small map. Concretely, they showed the demo about 3 machines in a 10×10 grid map. As the flow chart in their paper shows, the algorithm depends on the condition statements. As a result of that, it cannot perform with all the dispensable computation effort of the computer.

As the development of information technology, Štefanič provides an overview of emerging smart construction applications in areas such as construction monitoring, construction site management, safety at work, early disaster warning, and resources and assets management [32]. Also, Tumer describes the future construction site utilizing industry 4.0 [33]. Without a doubt, a future working site should be fully benefited from AI [34]–[36], automation technology [37], Simultaneous Localization and Mapping (SLAM) [38], and IoT [10], [11], [39], [40]. Therefore, the uncertainty degree of construction sites is reducing, and thus we consider the construction sites as known in our research.

Based on the strict literature review, A^* is the most developed and latest algorithm to solve construction sites' path tasks. It combines both step-cost calculation from the Dijkstra algorithm and feedback step from the genetic algorithm. However, as the number of machines increases, naive A^* might not be suitable to solve the MAPF problem due to algorithm complexity. Thus, it is necessary to explore the SOTA solutions in the field of mobile robotics, see next subsections, in order to find a more appropriate solution.

C. MAPF FOR MOBILE ROBOTICS

For the single agent, the planning task can be described as finding the lowest cost from the starting point to the targeting point. By using Heuristic Search, e.g., A^* Algorithm, such problem can be better solved. However, naive A^* Algorithm only considers that all the obstacles are static, which is the ideal assumption in the pathfinding problem [41]. In contrast, to solve the MAPF problem, we need to consider the other participants in the map, i.e., the dynamic obstacles also affect the optimal path of an individual agent.

MAPF for mobile robotics is both a well-studied and dynamic developing topic. The usage scenarios of MAPF and their corresponding algorithm are diverse, such as warehouse [9], computer games [42], and autonomous driving in intersection [43]. Until the end of 2020, although the concept of reinforcement learning is attracting more and more attention, current influential research about the shortest path and MAPF is mainly based on graph theory. To date, there is no universal solution for all kinds of pathfinding problems; thus, algorithms with different time and space complexities are proposed [44].

Based on the survey paper from Felner *et al.* [45], we know that no algorithm dominates all others in all circumstances. There is a tradeoff between high-quality path solutions and

realtime performance. The mainstream MAPF solution can be classified into search-based solvers and rule-based solvers. The former intends to find the best solution or near-optimal solutions, whereas the latter can run much faster, however, produce far away from optimal solutions. Of course, some compromised solutions combined two ideas together, namely, hybrid approaches. Another type of solution, namely reduction-based optimal solvers, focusing on reducing MAPF problems into some problems, such as the Constraint Satisfaction Problem (CSP), with a well-known solution. Since this approach usually only aims at the makespan tasks, we will not go much deeper in this approach. To date, the most influential solutions for MAPF are Conflict Based Search (CBS) and its variants due to their widely used real-world applications.

Sharon proposes CBS, combining both advantages from coupled and decoupled approaches. Although the pathfinding process is strictly single-agent searches, it can guarantee to offer optimal results, unless the one variants which deliberately provide a suboptimal solution for the purpose of realtime performance. As the authors introduce, CBS adopts a two-level structure, where the high-level search can be described as a Constraint Tree (CT), including every constraint. Then, the lower level finds the concrete path for each agent individually with the information from the higher level. The brilliance of this design is that the search process is not more exponential in the number of participants but exponential in the amount of conflicts encountered during the pathfinding process.

Since CBS tries to find the optimal solution and thereby cause a relatively longer runtime. In improved CBS algorithm [46], Boyarski summaries two methods to reduce the runtime. Concretely, it firstly adopts the Meta-Agent CBS (MA-CBS) [47], which merges multi-agents together and handles them as a large agent. Moreover, it uses bypass improvement, which encourages one of the agents to find an alternative path instead of performing a split at the high level. As mentioned by Boyarski, the bypass concept successfully avoid the unnecessary generate the new nodes in the CT [46]. Since the bypass concept only tries to find the solution from the path with the same cost as the one that shall be replaced, the optimality cannot be harmed. In the high-level search, Felner suggests adding heuristics into CBS so that the conflicts are not arbitrarily chosen [48]. After that, Li found the improved heuristics to guide the high-level search [49]. Also, she introduced the CBS with disjoint splitting [50]. The main contribution of CBS with disjoint splitting is the novel terminology of positive constraints forcing the a_i to be at v at timestep t. In this fashion, CBS with disjoint slitting reduces the amount of unnecessarily expanding the CT. In addition, some improvements, such as Lazy CBS, which avoids the behavior that CBS resolves the same conflicts between the same pairs of agents many times owing to lack of connection among subproblems [51]. Hönig proposed an approach called Conflict-Based Search with Optimal Task Assignment (CBS-TA) [52]. The improvement is mainly because it creates

the forest on demand. Solving MAPF optimally is proven to be NP-hard, so CBS and all other optimal solvers do not scale up. Alternatively, Barer proposed a suboptimal variant of the CBS algorithm [53] so that the problem can be solved suboptimally but much faster.

To sum up, naive CBS is an optimal pathfinding solution that is based on graph theory. The time consumption of the algorithm mainly depends on the conflicts occurring among the agents since they increase the nodes in the high-level tree. Its performance on bottlenecks and corridors is better, whereas the performance on open space can be worse than enhanced A^*. Thus, CBS's variants are focusing on reducing the nodes in the CT and therefore let CBS has a higher success rate in general. Note that in this paper, we use the same terminology as Stern's research to avoid misunderstanding; however, some mathematical descriptions might be adjusted.

III. PROBLEM STATEMENT

Although the path planning problem has been attracted engineer's attention, the proposed method is quite tricky to be used in a large construction site with many machines. Theoretically, given a 4 neighborhood movement model and an undirected graph $G(V, E)$, the branching factor should be estimated as $(E/V)^k$ and the search space is V^k if k machines should be planned. For 20 agents, considering that a normal working site with $500m \times 100m$ where 50×10 cells are needed, the search space goes to 9.54×10^{53}, which is unsolvable within acceptable duration for a real application even if the top CPU in 2020 achieving approaching 200 GFLOPs is used. Thus, using the traditional methods to solve the cooperation task is still challenging. Also, some algorithms are based on replanning the path if agents encounter a head-to-head position. However, these methods require excellent perception capability and limit the movement velocity of agents.

On the other hand, CBS based solution treats all the objects equally. Concretely, each robotics has the same capability, the cell on the ground is assumed as equally challenging to be overcome. However, it does not hold true in a working site; some machines should be assigned a higher priority, and some paths are much easier to pass. For instance, larger machines are usually more challenging to control their velocity. Also, stopping on a slope is much dangerous than on the flat ground. Hence, in our study, we further develop the original CBS so that it can deal with plenty of priority problems in a real working site. Also, the computational time should be much shorter to handle emergence.

IV. MODEL BUILDING

Nowadays, with the development of SLAM regarding visual recognition, IoT, and satellite technology, the uncertainty degree of construction sites is reducing. Thus we consider the construction sites as known in this paper. In most instances, the MAPF solution can be evaluated twofold. The first one is sum-of-costs which describes the accumulative cost of all the agents. Such costs can be time consumption, fuel
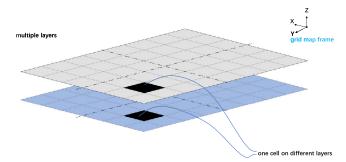

FIGURE 2. An example of multilayered grid maps. Our approach depends on multilayered grid maps to offer data of different types of information to make the best path. Concretely, every grid saves a 1*3 matrix, including location information and the corresponding terrain information. The map, which can be visualized in the BIM system, is saved as a 2*m*n*3 tensor, where m is the max displacement in the x-direction, while n is the max displacement in the y-direction. In case a place is unknown, it will be marked as NaN to denote the uncertainty of the regions and be treated as obstacles.
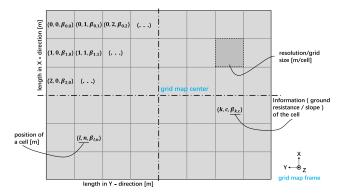

FIGURE 3. Detail description of a layer in the grid-based map.

consumption, or some other objective goals. The other way to analyze the performance of MAPF solvers is makespan, indicating the maximal time the last goal has been achieved. Obviously, unlike the robotics in warehouses, working machines perform a relatively long period to do their duty after they have arrived. Thus, makespan is not so vital compared to sum-of costs in the field of construction or milling machines. Consequently, we adopt sum-of-cost as our evaluation criterion rather than makespan.

For working site MAPF problems, we can describe the problem as given a graph, $G(V, E)$, and a set of k agents labeled as $a_1 \dots a_k$. Each agent a_i has a start position $s_i \in V$ and goal position $z_i \in V$. Based on the practical conditions in a working site, we consider vertex conflicts, edge conflicts, and swapping conflicts as unacceptable conflicts, whereas following and cycle conflicts are allowed in our study. Formally, the vertex and swapping conflicts are described as,

$$\begin{cases} \pi_i[t] = \pi_j[t], \\ \pi_i[t] = \pi_j[t+1] \cup \pi_i[t+1] = \pi_j[t] \end{cases} \quad (1)$$

where π_i and π_j are the single-agent path for a_i and a_j at time step t, correspondingly. The first equation shows the vertex conflict whilst the second equation describes the swapping
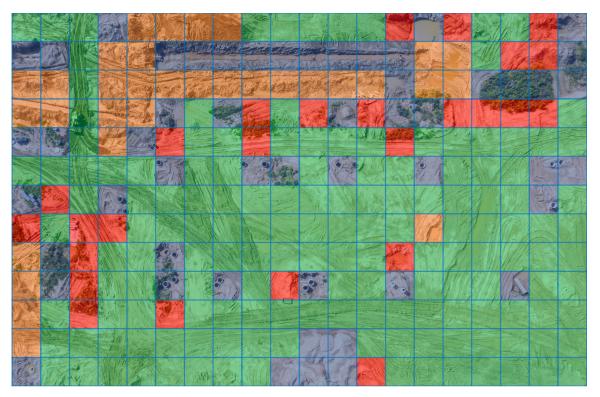
FIGURE 4. A grid map with terrain weight based on a real construction site. In this map, the green, orange, and red grids demonstrate the easy, normal, and difficult to pass terrain, separately. The blue cells denote the place where it is considered impossible to pass, such as occupied by the obstacles. On a real construction site, the obstacles can be the place to store construction materials temporarily.

conflict. Apparently, edge conflict is a union of vertex conflict. Intuitively, $\pi_1[2]$ denotes the location of the first agent at second time step.

A. MULTI-LAYER GRID MAP

Unlike a standard graph problem which adds the weight on the edges, we add the weight directly on the grid. This is mainly for three reasons. First and foremost, the machines usually occupy a relatively large area and thus should not be modeled as a simple vertex and ignore their geometry. Also, most previous studies in the field of construction machines used the grid-based map. To guarantee compatibility, we tend to use a similar solution since no approach obviously outperforms the other one. Last but not least, if weights are applied on edges, it becomes challenging to penalize the waiting process.

Inspired by the research from Fankhauser and Hutter [54], a map can include many layers to store different types of data information. Obviously, the map information should be saved in BIM system so that the path planning process can be done. In a previous study [38], Xiang developed a realtime map plotter of the construction site according to ground condition, offering multi-layer grid-based maps, which divide the environment into uniform cells. Moreover, maps can also be gathered by Lidar or cameras provided depth information installed on a drone or on the ground.

Fig. 2 illustrates the multilayered grid map concept, where each cell data is stored on the congruent layers. In many

construction projects, since resistance and grade of the road are the most of importance information for the construction machines, we show a two layers grid map as an example. Concretely in our study, the map is divided into small cells, whose resolution is 10 meters per cell to cover the geometry of the vehicles. In practice, we can use GPS/IMU based Kalman filter algorithms to locate the mobile machine in the construction site. As can be seen in Fig. 2, to describe the ground condition of construction sites, we use the value of each cell to represent the information of the ground situation. In Fig. 3, a layer that holds the data of a grid-based map is shown. Apparently, although we only demonstrate the map with two layers, it is relatively easy to extend the third layer in case more information should be taken into account for the path planning because we can simply add the weights together.

B. LOWER LEVEL SEARCH

The concept of CBS does not limit the lower-level search algorithm. In addition, since negative weight cannot happen in 3D space, we believe that Dijkstra and A^* can work well for individual shortest path search. In light of that, we use best-first search. Also, to accelerate our algorithm, we limit the moving direction of an agent to 4 and thus reduce the branching factor to 5, including wait, instead of 9 or more. In order that we can get the optimal solution, we set our heuristics smaller than the real distance since weights are
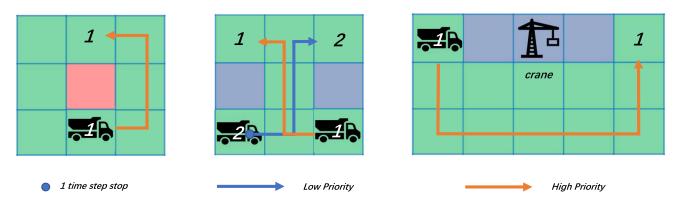
FIGURE 5. The basic requirements of the path planning algorithm. As shown in the left subfigure, the vehicle should take the lowest-cost path to reach its goal. The middle subfigure shows that the vehicle with lower priority should wait until the vehicle with higher priority pass through if there is no other bypass possibility. Last but not least, the right subfigure indicates that a good path should not be too close to dangerous objects. Girds in blue, red, and green represent obstacles, rough road, and normal road separately.

considered. Because grid map is used, we use Manhattan distance as the base of our heuristics to guide our search.

In the lower level, the algorithm searches the best path of individual agent based on the estimated cost of through current vertex to the goal, formally,

$$f_{f,r}(n) = \sum_{i=1}^{p} W_{L,i} \cdot g_{i,f,r}(n) + h_{f,r}(n) \quad (2)$$

where $f(n)$ is the estimated cost from its source through current vertex n to its goal, g_i denotes the real cost from the source to the current vertex n considering the i_{th} weight-grid map, and h denotes the estimated cost from the current vertex n to the predefined goal based on Manhattan distance. W_L is the weight for the specific layer. The index f and r show the estimated cost is forward or backward.

Apparently, planning the best path for machines to reach their goal is a multi-objective task. Generally speaking, the evaluation criterion can be divided into subjective and objective criteria. Obviously, the objective criterion demonstrates the objective criterion of the planned path, especially the terrain which can affect safety and efficiency. As some roads inside the construction site can be built with asphalt, so is considered better road conditions than some road made of sand. Consequently, the cost of passing different routes is different. Besides that, the road slope should be taken into account since waiting on a steep hill is much more dangerous than staying on flat ground. Therefore, we introduce multi-layer to record the individual characteristics of the terrain and plan the best path based on them. Concretely, a construction site map is divided into a series of cells and layers, according to different criteria. The weights of individual cells in one layer are saved as shown in the following matrices.

$$\tilde{W}_{m,n} = \begin{pmatrix} w_{1,1} & w_{1,2} & \cdots & w_{1,n} \\ w_{2,1} & w_{2,2} & \cdots & w_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ w_{m,1} & w_{m,2} & \cdots & w_{m,n} \end{pmatrix}$$

In contrast, the subjective criterion may not harm the whole system's actual performance; however, it has an impact on people's psychology. For instance, a crane or some other dangerous objects, such as a power station, on a working site should be protected, and we should avoid the mobile machines unnecessarily approaching them. As we know, even machines did not involve in an accident, getting close to a dangerous object will be stressful for site managers and indicating a potential risk. To address this problem, we add g_3 to penalize the machines for occupying the areas surrounding these special objects.

$$g_3(n) = \sum_{o=1}^{r} \frac{C_o}{|(X_n - X_o)| + |(Y_n - Y_o)|} \quad (3)$$

where $[X_o, Y_o]$ is the position of the objects which should try to avoid being approached, C_o denotes the intensity.

To accelerate the low-level searching process, we utilize the bidirectional A^* algorithm with a path return for the initial search. The following Algorithm 1 shows the algorithm's steps, where CF is the dictionary to save the path sequence. Since bidirectional A^* is not suitable for dealing with the waiting operation, neighbors are valid if the grids are not occupied by obstacles, i.e., ignoring the conflicts with other agents. The index 0 in the algorithm denotes forward, whereas index 1 means backward.

After the initial solution is proposed, we adopt unidirectional A^* to solve the conflicts among the agents. Also, the priority of the agents is taken into account here. As a conflict occurs, the algorithm will give the order which agents should avoid other higher priority agents. Unlike some other research using hard priority, which might lead to the algorithm become not complete, we adopt the soft priority, namely penalization function, to guarantee the algorithm to find the feasible solution if the problem is a solvable MAPF problem. Since the A^* algorithm is well known, we only give the part

Algorithm 1 Bidirectional A^* Algorithm at Low Level to Speed up the Searching Process

---

Input: $G(v, t)$, $\tilde{s}$, $\tilde{z}$ from predefined map information in yaml, original from visual or Lidar recognition

Output: *Path*, $d_{shortest}$

    Initialisation:

1:   $OpenSet \leftarrow [s, z]$; $ClosedSet \leftarrow [\Phi, \Phi]$

    $G^R \leftarrow ReverseGraph(G)$

2:   $dist[:] \leftarrow \infty$, $dist^R[:] \leftarrow \infty$

3:   $dist[s] \leftarrow 0$, $dist^R[z] \leftarrow 0$

4:   $[CF, CF^R, proc, proc^R] \leftarrow \Phi$

    LOOP Process

5:   **while** $OpenSet[0]$, $OpenSet[1]$ not *empty* **do**

6:      $v_c \leftarrow ExtractMin(dist)$, forward otherwise $dist^R$

7:      $OpenSet.remove[0](u)$

8:      **if** neighbor(u) = valid **then**

9:        **if** u not in $ClosedSet[0]$ **then**

10:          $g[0][u]_{tmp} \leftarrow g[0][v] + StepCost$

11:        **end if**

12:        **if** neighbor not in $OpenSet[0]$ **then**

13:          $OpenSet.append(u)$

14:        **else if** $g[0][u]_{tmp} > g[0][u]$ **then**

15:          continue

16:        **end if**

17:        $CF[0][u] \leftarrow v$

18:        $pi_f \leftarrow |\overrightarrow{d(v) - d(s)}|$, $pi_r \leftarrow |\overrightarrow{d(t) - d(v)}|$

19:        $h_f \leftarrow (pi_f - pi_r)/2$

20:        $h_r \leftarrow -p_f$

21:        $f[0][u] \leftarrow g[0][u] + h_f$, if forward otherwise h_r

22:      **end if**

23:      **if** u in $ClosedSet[0]$ **then**

24:        break

25:      **end if**

26:      $ClosedSet[0].append(u)$

27:      repeat symmetrically for v^R as for v, where $OpenSet[1]$ and $ClosedSet[1]$ should be used

28: **end while**

29: $distance \leftarrow \infty$, $u_{best} \leftarrow None$

30: **for** u in $ClosedSet[0] + ClosedSet[1]$ **do**

31:      **if** $dist + dist^R < distance$ **then**

32:        $u_{best} \leftarrow u$

33:        $distance \leftarrow dist[u] + dist^R[u]$

34:      **end if**

35: **end for**

36: $last_0$, $last_1 \leftarrow u_{best}$

37: **while** $last_0 \mathrel{!}= s$ **do**

38:      path.append(last)

39:      $last_0 \leftarrow CF[0][last_0]$

40:      path.reverse

41: **end while**

42: **while** $last_1 \mathrel{!}= t$ **do**

43:      path.append(last)

44:      $last_1 \leftarrow CF[1][last_1]$

45: **end while**
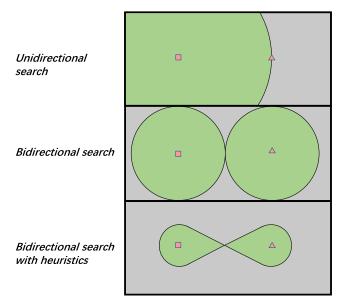
46: **return** *path*, *distance*

FIGURE 6. Illustration of the benefit of bidirectional search with heuristics. We use squares and triangles to represent the start point and the goal point separately. Here the grey region denotes the whole region of the map, and the green region shows the region that the algorithm must search before finding the shortest path for agents. Apparently, bidirectional search with heuristics cannot be slower than its antecessors.

where involves in the priority of the agents, in Eq. (4),

$$g[u]_{tmp} = g[v] + P_{a_i} \cdot \sum_{k=1}^{L} StepCost \qquad (4)$$

where u is the neighbour point and v is the current point. L is the total layers of the map, and P_{a_i} is the priority value of the agent *i*.

C. HIGHER LEVEL SEARCH

Although the papers about CBS and its variants showed the success rate of the algorithms under various specific scenarios, they are performed with a time limit of at least 1 minute. Considering that some machines might not maintain their speed and emergencies may happen, we believe a feasible algorithm for the construction site should give a command to all the participants within 5 seconds even the order can be just wait.

Concretely, we let some machines have priority to move to their goal while the others should wait for a while or find a midway destination in case when the task is too complicated for a realtime response. Or the algorithm suggests to reduce the total mount of machines on site if necessary. Different from the original CBS algorithm, we add four different strategies in case the algorithm cannot find a solution for all agents within 5 seconds. The basic ideas of these acceleration process are reducing the complexity of the task by solving the problem step by step and described as follow,

- Directly remove the agents causing most conflicts, including initial conflicts and update conflicts, found by

Algorithm 2 High Level Realtime Search

Input: $G(v, t), \tilde{s}, \tilde{z}$ from predefined map information in yaml, original from visual or Lidar recognition

Output: solution for all agents, total cost

    Initialisation:

1:   *start.constrains* $\leftarrow \Phi$

2:   *start.solution*, *start.cost* $\leftarrow$ *bid_Astar.search()*

3:   *allConflicts* $\leftarrow$ *findConflicts(start.solution)*

4:   insert start to OPEN

    LOOP Process

5:   **while** OPEN not Empty **do**

6:      $P \leftarrow$ the node with lowest solution cost

7:      **if** *counter* > *threshold* **then**

8:         *P.remove(allConflicts)* (remove some agents)

9:      **end if**

10:     **if** Validate(P) = 1 (no conflicts found) **then**

11:        return $P.solution$, $P.cost$

12:     **end if**

13:     $C \leftarrow$ first conflict (a_i, a_j, v, t) in P

14:     *allConflicts.append(C)*, *counter* ++

15:     **for** a_c in C **do**

16:        $ND \leftarrow P$

17:        *ND.constraints.append(a_c, v, t)*

18:        *ND.solution.update(uni_Astar.search())*

19:        *ND.cost.update(SIC(ND.solution))*

20:        Insert ND to OPEN

21:     **end for**

22: **end while**

---

line 3 and line 13 in Algorithm 2. Since the computation time of the CBS-based algorithm depends on the conflicts number, reducing the trouble makers can surely speed up the searching process.

- Let the machines have the same moving direction to move first. Obviously, conflicts can be avoided if all the agents move in the same direction.
- Randomly select some machines in independent sub-regions to move first.
- Let the machines having lower estimated cost move first.

The individual path will be compared at the high-level search to find out the conflicts among the agents. We use bidirectional A^* algorithm with the heuristics proposed by [55] to create the initial path of each agent in order to enhance the realtime performance, and afterward utilize unidirectional A^* to update the individual path of each agent since only unidirectional A^* can deal with the waiting process. In the BIM system, during the searching process, the algorithm saves the conflicts position and the corresponding agents. As mentioned, in case the algorithm cannot solve the planning problem due to too many conflicts, the algorithm will remove some agents and then replan the paths of the rest agents. In this fashion, we ensure that the known MAPF problems can be solved in a timely manner. If emergence occurs and the algorithm cannot solve the new task in time, we can easily and
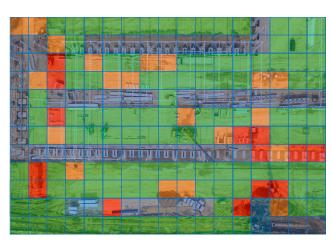

FIGURE 7. Map example. Another map based on a real construction site which has more narrow corridor.

TABLE 1. Weight table. The weights we use to describe the complicated terrain of construction sites.

Layers	Weight
Roughness	[1, 5, 9]
Slope	[1, 5]
Safety	[1, 15]

quickly locate the trouble maker. Apparently, the optimization direction here is not only to ensure a short calculation time, but also to make as many machines as possible move at the same time.

V. EXPERIMENT ON REAL WORKING SITES

As a consensus of the research in the field of graph theory, although a conclusion about a specific map cannot guarantee its effectiveness on another map, the closer the map, the closer the effect. In order to show the benefit of the introduction of MAPF in the mobile machines industry, we validated our algorithm on five typical real working sites. Concretely, they are a relatively open field with 20 or 50 agents, an open field with many obstacles, a two-side working site connected by a bridge or narrow corridor, and a typical mining site. Since the map will also be shown in the path results as background, here we only demonstrate how a map will be processed to give the prior information for successfully pathfinding on the first map and third map to avoid repetition. The maps shown in Fig. 4 and Fig. 7 are on the same site at different times. Obviously, Fig. 4 is the earlier stage while Fig. 7 shows the later stage as the construction process proceed since more facilities are there. In our experiments, the dimensions of our maps are 20×13 and 17×12. For the sake of simplicity, we also assume the velocity of all the machines are constant; however, it is surely easy to achieve the situation that the machines have quite different speed since we can use the fastest speed as a reference and allow the slower agents occupy more than one grid at the same time or vice versa.

As we can imagine, the faster the project, the faster the construction site changes. This indicates the difficulties of
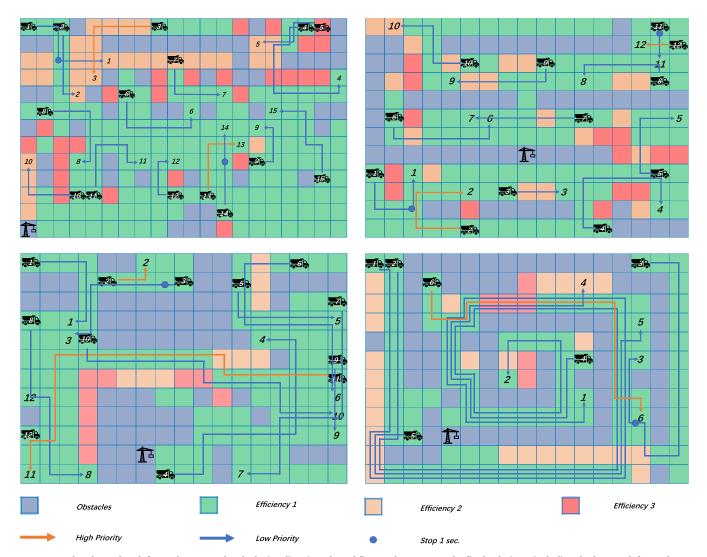
FIGURE 8. The planned path for each map. In the clockwise direction, the subfigures demonstrate the final solutions, including the best path for each machine for maps 1, 3, 4, 5. The left bottom point is defined as the original point (0,0), and the horizontal axis is the first axis. The layout of map 2 is the same as map 1; however, the difference is that there are more agents on map 2. Due to its huge amount of information, we give the planned schedule in Tab. 3 instead of using figures.

using a pre-calculated path planning for a construction site. In this study, we divided the grids into different regions with respect to whether the road is easy to be passed through, the slope of the road, and whether the place is safe.

Here we show the solution finding time on CPU core i7 4720HQ@ 2.6GHz. Because of its reasonable price at the end of 2020, it is suitable for large-scale commercial use. To reduce the randomness, we did the experiments 50 times and gave the average finding time, and the average number of conflicts occur to analyze the conflicts and thus show the rationality of our optimization. Notice that we rounded the numbers to one decimal place if any.

Before we analyze the results of our experiments, we summarize the basic ideas of the algorithm we used. Similar to the original CBS algorithm, our MAPF algorithm also adopts a two-level search, where the upper level finds the

conflicts among the agents and the lower level search the best path for individual agents. The lower level finds the path first and sends the initial proposal to the upper level. Afterward, the upper level will check whether the planned path has a or many conflicts with others. In case there are no conflicts, the center commander, an AI system, agrees to the preliminary proposal to become the final solution of MAPF and all agents are allowed to execute this solution. In other cases, if there have some conflicts, the upper level will find out the conflicts and send this information as constraints to the lower level to avoid the conflicts. To generate the initial individual path for each machine faster, we use bidirectional search. And then update the individual path if the upper level finds out a conflict with unidirectional A^* algorithm. The algorithm tries to modify the solution having the lowest cost, which guarantees the solution to be optimal. In the experiments, we do not assume what the participants are, nor
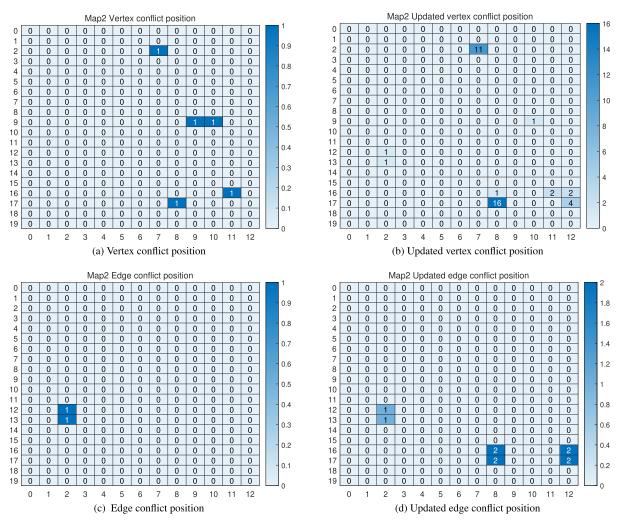
(a) Vertex conflict position

(b) Updated vertex conflict position

(c) Edge conflict position

(d) Updated edge conflict position

FIGURE 9. The place where the agents intend to pass through and the resulting conflicts on the second map. As we can see from the points allocation, initial conflicts have a great relevance to the conflicts occurring during the conflicts avoidance process. Notice that we did the experiment 50 times and the conflicts shown in these figures are the average number of these 50 experiments. Intuitively, the update process is to find a bypass for the agents who have conflicts with others. However, this process may cause other conflicts with other agents. For example, to avoid conflict with agent 1, the new schedule of agent 2 may overlap with agent 3.

do we assume its working process to ensure the generalization of our method.

VI. EXPERIMENT AL RESULTS

In this section, we demonstrate the planned path for each map in Fig. 8. As we can see, our algorithm successfully finds out the optimal paths considering the priority of the machines, i.e., the path with the lowest cost considering the main criterion, for all the tested maps. The algorithm commands the machines to drive directly to the goal, find a bypass, or just wait for others first to pass through.

In Tab. 2, we give the computational time to find out the optimal solution. We divide the searching time into initial search and the following update process. In the first phase, the computational time is no more than 0.1 seconds on our tested maps. In case that the MAPF task is easy, i.e., the counteraction and potential conflicts among the agents are rare, the update process can also be done very fast. As we

can see, the total duration to get the optimal solution is within 0.2 seconds for the scenario of agents on map 1. However, in other cases, such as the MAPF tasks on map 2 and map 4, although the period to offer the initial path proposal has no significant difference, the total duration is quite different. Concretely, the tasks on map 2 and map 4 need about 6.8 and 10.4 seconds to be solved. For such tasks, we can of course find the solution inside the BIM system before the machines execute their order saved in the schedule file. In the ideal case, the computational time for finding the solution is not the critical thing. However, in a real application, it is normal that the participants do not act on time when something urgent happens; thus, the ability to replan the path quickly is particularly important rather than let all the machines wait in place. As shown in Fig. 11 and Fig. 12, the update process is a dominant part of the whole searching process. Also, with the data shown in Tab. 2, comparing the duration of the initial search and the following update process, we can conclude
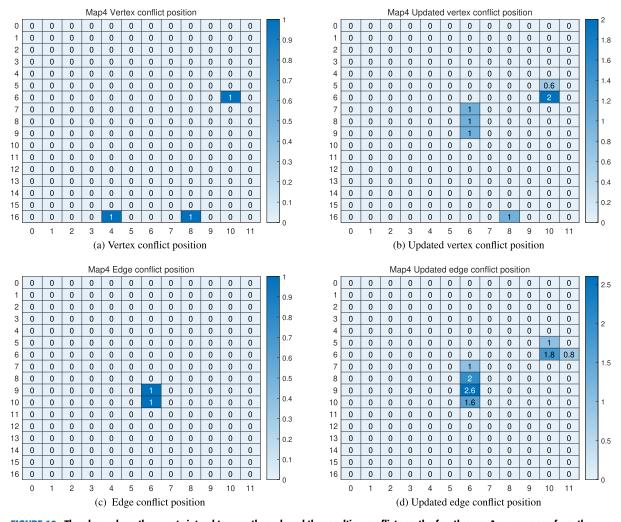
(a) Vertex conflict position

(b) Updated vertex conflict position

(c) Edge conflict position

(d) Updated edge conflict position

FIGURE 10. The place where the agents intend to pass through and the resulting conflicts on the fourth map. As we can see from the points allocation, initial conflicts still have a great relevance to the conflicts occurs during the conflicts update process.

(a) Edge conflicts and updated edge conflicts

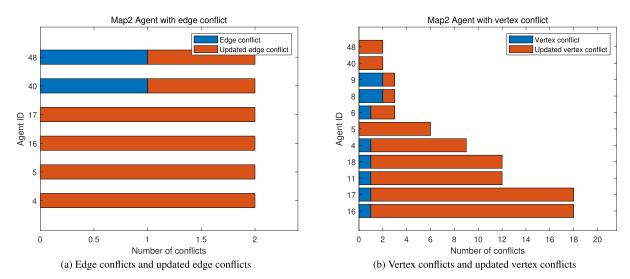(b) Vertex conflicts and updated vertex conflicts

FIGURE 11. Statistics of the conflicts made by corresponding agents on map 2. Here blue histogram denotes the conflicts found by initial bidirectional search, and the orange histogram shows the conflicts solved while updating the solution with unidirectional search.
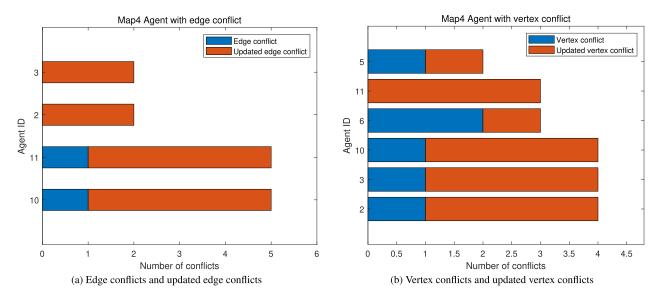
(a) Edge conflicts and updated edge conflicts

(b) Vertex conflicts and updated vertex conflicts

FIGURE 12. Statistics of the conflicts made by corresponding agents on map 4.

TABLE 2. The average algorithm searching duration (s). "None" denote that there is no need to optimize since the response is already quick enough.

Map Nr.	Map1	Map2	Map3	Map4	Map5
Bidirectional A* algorithm duration	0.0302	0.0844	0.0178	0.0400	0.0322
Update algorithm duration	0.1468	6.8192	0.1680	10.4038	1.3689
Theoretical cost before optimization	648.9523	1317.0138	492.3904	435	1066
Total duration by layout optimization	None	0.3058	None	0.1746	None
Total duration by agent optimization	None	0.9726	None	0.1731	None
Total duration emergence	None	0.4200	None	0.1489	None

that reducing the update process is the main optimization direction to make our algorithm faster.

In this paper, we demonstrate the optimization mechanism on the MAPF task on map 2 and map 4 to avoid wordy; however, we confirm the conclusions we make are also in line with the other maps we tested. The approaches were also tested on some other maps while only the results on map 2 and map 4 are shown in Tab. 2. The optimization depends on the stage of the construction site. In the early state before the site is built up in reality, we have more freedom to optimize. To accelerate the computation, two methods are proposed for the early stage. The first idea is to modify the unreasonable part of the construction sites. In this fashion, we can improve the throughput of the construction site. Fig. 9 represents the positions where the algorithm commands the machines to pass through but encounters conflicts with other construction machines. As aforementioned, we consider two kinds of conflicts in this study since they are more in line with the construction site, namely edge conflicts and vertex conflicts, respectively. Fig. 9 (a) and (c) demonstrate the conflicts found by initial bidirectional search. Since the map is weighted, the best path is usually unique; this is partly proved by the fact that the conflicts number found by initial bidirectional search is a multiple of the time we did the experiments. However, the unique best solution increases the possibility of generating conflicts. Taking the MAPF task on

map 2 for example, as shown in Fig. 9, we can see there are three regions that have more conflicts than others. Concretely, they are the region including vertex (16, 8) and (17, 8), the region including vertex (2,7), as well as the region including vertex (17,12) and (16,12). Correspondingly, based on the intended movement of the agents around these positions, the algorithm points out that the vertex (16,9), (17,9), (3,8), and (15,12) shall be modified to have similar characteristics as its surrounding. For instance, the road condition of the vertex (16,9) and (17,9) shall be changed into good condition from bad condition since their surroundings have good condition. The computational time is then dramatically reduced to about 0.31 seconds and seems to be the most effective method to reduce the solution finding time. The results are demonstrated in Tab. 2.

The other idea is to remove the most troublemaker in the MAPF task. As we know, the capacity of each construction site has a physical upper limit. No matter how excellent the algorithm is, too many participants will eventually lead to a decline in overall performance. Compared to the first method, which has a potential drawback that there might be some reasons that optimization of the working site is not always feasible, removing a conflicts-causing agent can be used whenever needed. In the case of the MAPF task on map 2, we can remove agent 16 according to Fig. 11 so that the computational duration reduces to roughly

TABLE 3. The schedule of MAPF task on map 2. In this task, the algorithm should plan the path for 50 agents.

agent name	start	1	2	3	4	5	6	7	8	9	10
agent1	[0, 12]	[1, 12]	[2, 12]	[2, 11]	[2, 10]	[3, 10]	[4, 10]				
agent2	[2, 12]	[2, 11]	[3, 11]	[4, 11]							
agent3	[6, 12]	[7, 12]	[8, 12]	[9, 12]	[10, 12]						
agent4	[17, 12]	[16, 12]	[16, 12]	[16, 11]	[15, 11]	[15, 10]	[15, 9]				
agent5	[18, 12]	[17, 12]	[17, 12]	[16, 12]	[16, 11]	[15, 11]	[15, 10]				
agent6	[14, 11]	[15, 11]	[16, 11]	[17, 11]	[18, 11]						
agent7	[0, 10]	[1, 10]	[2, 10]	[2, 9]							
agent8	[8, 10]	[8, 10]	[9, 10]	[9, 9]	[10, 9]						
agent9	[10, 10]	[9, 10]	[9, 9]	[8, 9]							
agent10	[0, 9]	[1, 9]	[2, 9]	[2, 8]	[2, 7]	[3, 7]					
agent11	[2, 9]	[2, 9]	[2, 8]	[2, 7]	[3, 7]	[4, 7]					
agent12	[6, 9]	[6, 10]	[7, 10]	[8, 10]							
agent13	[12, 9]	[12, 8]	[12, 7]								
agent14	[19, 9]	[19, 8]	[18, 8]	[17, 8]	[16, 8]	[15, 8]	[14, 8]				
agent15	[5, 8]	[6, 8]	[7, 8]	[7, 7]	[7, 6]	[8, 6]	[9, 6]	[10, 6]	[10, 7]	[10, 8]	[11, 8]
agent16	[15, 8]	[16, 8]	[16, 7]	[17, 7]	[17, 8]	[17, 9]					
agent17	[19, 8]	[18, 8]	[17, 8]	[16, 8]	[15, 8]	[14, 8]	[13, 8]				
agent18	[0, 7]	[1, 7]	[2, 7]	[2, 6]	[2, 5]						
agent19	[1, 7]	[2, 7]	[3, 7]	[4, 7]	[4, 6]	[4, 5]	[4, 4]	[3, 4]			
agent20	[6, 7]	[7, 7]	[7, 6]	[7, 5]							
agent21	[7, 7]	[7, 6]	[8, 6]	[9, 6]	[9, 5]						
agent22	[13, 7]	[13, 6]	[14, 6]	[14, 5]							
agent23	[15, 7]	[15, 6]	[15, 5]								
agent24	[17, 7]	[17, 6]	[17, 5]	[16, 5]							
agent25	[4, 6]	[5, 6]	[6, 6]	[6, 5]							
agent26	[6, 6]	[6, 5]	[6, 4]								
agent27	[1, 0]	[2, 0]	[3, 0]	[4, 0]							
agent28	[7, 1]	[8, 1]	[8, 0]								
agent29	[13, 0]	[14, 0]	[15, 0]	[16, 0]	[17, 0]	[18, 0]	[19, 0]				
agent30	[17, 6]	[17, 5]	[18, 5]	[19, 5]	[19, 4]						
agent31	[19, 6]	[19, 5]	[19, 4]	[19, 3]							
agent32	[1, 5]	[0, 5]	[0, 4]								
agent33	[4, 5]	[4, 4]	[4, 3]								
agent34	[7, 5]	[8, 5]	[8, 4]	[8, 3]							
agent35	[8, 5]	[9, 5]	[10, 5]	[10, 4]							
agent36	[13, 5]	[12, 5]	[11, 5]	[11, 4]							
agent37	[17, 5]	[17, 4]	[16, 4]	[16, 3]							
agent38	[6, 4]	[6, 3]	[6, 2]								
agent39	[12, 4]	[12, 3]	[11, 3]								
agent40	[14, 4]	[14, 5]	[13, 5]	[12, 5]	[12, 4]	[12, 3]					
agent41	[19, 4]	[18, 4]	[17, 4]								
agent42	[0, 3]	[0, 2]	[1, 2]	[2, 2]							
agent43	[8, 3]	[8, 2]	[9, 2]	[9, 1]							
agent44	[15, 3]	[15, 2]	[15, 1]	[14, 1]	[13, 1]						
agent45	[18, 3]	[19, 3]	[19, 2]	[19, 1]							
agent46	[0, 2]	[1, 2]	[1, 1]	[2, 1]	[3, 1]	[4, 1]					
agent47	[4, 2]	[4, 1]	[5, 1]	[6, 1]	[6, 0]						
agent48	[9, 2]	[10, 2]	[11, 2]	[12, 2]	[13, 2]						
agent49	[13, 2]	[14, 2]	[15, 2]	[16, 2]	[17, 2]						
agent50	[19, 2]	[18, 2]	[17, 2]	[17, 1]	[16, 1]	[15, 1]					

0.97 seconds. Considering the holistic productivity is only marginally affected since there are still 49 machines that can work well in the working site, and the shorter duration endows the whole system the capability to deal with the emergence on site, this method is recommended if the construction site cannot be modified.

In the BIM system, we consciously make sure that the computational duration is within an acceptable period. However, this cannot guarantee all the potential conflicts for these MAPF tasks are removed. For the example of the MAPF task on map 2, in case that agent 16 does not catch up with the planned schedule and has a two-time step delay, the duration for replanning the solution for the whole fleet go to more than 5 seconds, even the construction site is modified in the early stage. Since we have already optimized the searching process in BIM so that the duration is shorter than 0.5 seconds and the only thing changed here was agent 16, we could quickly draw a conclusion that the expanding computational time is due to agent 16. Afterward, the algorithm checks the other agents' paths and finds a reachable grid, which other agents will not occupy, to be set as the new destination for agent 16 to avoid conflicts. Concretely, the new goal for agent 16 shall be the vertex (15,7), and the computational time is then only 0.42 seconds. Notice that agent 16 is not allowed to stop at its original place since it blocks the only way for agents 17 and 14 to reach their goal. In case that machine 16 totally lost its

mobility, the algorithm will ask every participant to stop and wait for the human intervention.

In the MAPF task on map 4, we use the same methods to optimize the computation time in order to demonstrate our solution's generalization capability when the terrain is more complex and there are fewer agents. As shown in Fig. 10, 2 regions have more conflicts than others, i.e., the region including vertex (8,6) and vertex (9,6), and the region including vertex (6,10). Same as the method, namely construction site optimization, used in the previous case, the vertex (5,10) is indicated, which shall be modified from an obstacle to a road with good conditions. In addition, the road condition of the vertex (9,5) and vertex (10,6) shall be changed into good condition from bad condition. After this optimization, the computation time is greatly reduced to roughly 0.17 seconds. For agent optimization, we shall remove the most conspicuous troublemakers in this task, which are agent 10 and agent 11, as shown in Fig. 12. Here we removed the agent 10 so that the computation duration reduces to about 0.17 seconds. Although the optimization results of these two methods in the MAPF task on map 4 are almost the same, we recommend layout optimization because only 12 agents were deployed in map 4. If an agent is removed, the overall productivity will be reduced more significantly compared to the previous scenario on map 2. In case that agent 10 did not run perfectly according to the planned schedule and had a delay of one-time step at the beginning, the computation duration for replanning can also exceed 5 seconds, even if the layout of the construction site was optimized with the first method. Our algorithm gave a new goal for agent 10, which shall be the vertex (7,4), and the computational time was afterward 0.15 seconds. According to the above results, our algorithm is also proven as effective for the task on map 4.

VII. ADVANTAGES OF OUR METHODS

In this study, our approach enables many machines to work simultaneously inside of the working site. Firstly, the method helps civil engineers to arrange the construction site before the site is setup. Our approach points out the positions where conflicts occur among the machines and thus indicates the place worthy of being modified. Moreover, our algorithm also helps the engineers to determine the reasonable number of machines in a working site, on the premise of using advanced algorithms. In addition, our algorithm schedules a conflict-free solution for the agents so that the agents can move confidently without hesitation. Last but not least, since the emergencies are inevitable, we design the system to replan the path solution in a very short period compared to the SOTA MAPF algorithm with only slightly increase the non-optimality of the solution.

VIII. CONCLUSION

In this paper, we presented an efficient and effective algorithm to calculate the path of a fleet of machines on a construction site. Considering the complicated terrain of a construction site, we endow our algorithm with the ability to handle the weighted maps. By testing our method on five different and diverse maps, our method successfully found the best path for a fleet including participants with different importance. By solving the MAPF problem for a construction site from both algorithmic and construction layout perspectives, we showed the benefits of our hybrid method, especially in reducing the computational time to handle emergencies. Based on our results, modify the unreasonable part is the most efficient fashion to speed up the searching process. Also, removing the agents which cause the most conflicts is always viable and can dramatically reduce the searching time but slightly reduce the whole productivity.

A. OUTLOOK

As we mentioned in the literature review, path planning is a fast developing and prosperous research field; thus, we did not fully consider all the improved methods for the initial version of our method. For instance, we did not add in the mega-agents concept, which merges the agents together based on some specific rules to reduce the conflicts and thus speed up the searching process. Since we confirm that our method can be combined with these methods, we expect the searching process to be accelerated further.

In addition, although Huoshenshan's working site proved the concept, the more machines invested, the faster is the project, which is also consistent with our subjective imagination, a comprehensive study on the quantitative relationship between the number of machines invested and the productivity of the working site is not done. We encourage the experts in civil engineering to propose some challenging scenarios and test our algorithm on them.

ACKNOWLEDGMENT

REFERENCES

[1] A. Borrmann, M. König, C. Koch, and J. Beetz, *Building Information Modeling*. Cham, Switzerland: Springer, 2018.

[2] K. Barlish and K. Sullivan, "How to measure the benefits of BIM—A case study approach," *Autom. Construct.*, vol. 24, pp. 149–159, Jul. 2012.

[3] Y. Liu, S. van Nederveen, and M. Hertogh, "Understanding effects of BIM on collaborative design and construction: An empirical study in China," *Int. J. Project Manage.*, vol. 35, no. 4, pp. 686–698, 2017.

[4] R. Volk, J. Stengel, and F. Schultmann, "Building information modeling (BIM) for existing buildings—Literature review and future needs," *Autom. Construct.*, vol. 38, pp. 109–127, Mar. 2014.

[5] S. Song and E. Marks, "Construction site path planning optimization through BIM," in *Proc. Comput. Civil Eng.*, Jun. 2019, pp. 276–369.

[6] A. R. Soltani, H. Tawfik, J. Y. Goulermas, and T. Fernando, "Path planning in construction sites: Performance evaluation of the Dijkstra, A*, and GA search algorithms," *Adv. Eng. Informat.*, vol. 16, no. 4, pp. 291–303, 2002.
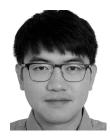
[7] A. Khalafallah and K. El-Rayes, "Automated multi-objective optimization system for airport site layouts," *Autom. Construct.*, vol. 20, no. 4, pp. 313–320, Jul. 2011.

[8] M. Yahya and M. P. Saka, "Construction site layout planning using multi-objective artificial bee colony algorithm with Levy flights," *Automat. Construct.*, vol. 38, pp. 14–29, Mar. 2014.

[9] H. Ma, W. Hönig, T. S. Kumar, N. Ayanian, and S. Koenig, "Lifelong path planning with kinematic constraints for multi-agent pickup and delivery," in *Proc. AAAI Conf. Artif. Intell.*, vol. 33, 2019, pp. 7651–7658.

[10] Y. Xiang, T. Su, C. Brach, X. Liu, and M. Geimer, "Realtime estimation of IEEE 802.11p for mobile working machines communication respecting delay and packet loss," in *Proc. IEEE Intell. Vehicles Symp. (IV)*, Las Vegas, NV, USA, Oct. 2020, pp. 1516–1521.

[11] Y. Xiang, B. Xu, T. Su, C. Brach, S. S. Mao, and M. Geimer, "5G meets construction machines: Towards a smart working site," 2020, *arXiv:2009.05033*. [Online]. Available: http://arxiv.org/abs/2009.05033

[12] K. Whitlock, F. H. Abanda, M. B. Manjia, C. Pettang, and G. E. Nkeng, "Bim for construction site logistics management," *J. Eng., Project, Prod. Manage.*, vol. 8, no. 1, p. 47, 2018.

[13] J. H. Woo, "BIM (building information modeling) and pedagogical challenges," in *Proc. 43rd Nat. Annu. Conf. (ASC)*, 2006, pp. 12–14.

[14] S. Azhar, "Building information modeling (BIM): Trends, benefits, risks, and challenges for the AEC industry," *Leadership Manage. Eng.*, vol. 11, no. 3, pp. 241–252, 2011.

[15] F. P. Rahimian, S. Seyedzadeh, S. Oliver, S. Rodriguez, and N. Dawood, "On-demand monitoring of construction projects through a game-like hybrid application of BIM and machine learning," *Autom. Construct.*, vol. 110, Feb. 2020, Art. no. 103012.

[16] R. Sacks, M. Girolami, and I. Brilakis, "Building information modelling, artificial intelligence and construction tech," *Develop. Built Environ.*, vol. 4, Nov. 2020, Art. no. 100011.

[17] S. Tang, D. R. Shelden, C. M. Eastman, P. Pishdad-Bozorgi, and X. Gao, "A review of building information modeling (BIM) and the Internet of Things (IoT) devices integration: Present status and future trends," *Automat. Construct.*, vol. 101, pp. 127–139, May 2019.

[18] G. Sullivan, S. Barthorpe, and S. Robbins, *Managing Construction Logistics*. Hoboken, NJ, USA: Wiley, 2011.

[19] S. H. Bengtsson, "Coordinated construction logistics: An innovation perspective," *Construct. Manage. Econ.*, vol. 37, no. 5, pp. 294–307, May 2019.

[20] E. R. Azar and V. R. Kamat, "Earthmoving equipment automation: A review of technical advances and future outlook," *J. Inf. Technol. Construct.*, vol. 22, no. 13, pp. 247–265, 2017.

[21] M. Hirayama, J. Guivant, J. Katupitiya, and M. Whitty, "Path planning for autonomous bulldozers," *Mechatronics*, vol. 58, pp. 20–38, Apr. 2019.

[22] M. Hirayama, J. Guivant, J. Katupitiya, and M. Whitty, "Artificial intelligence in path planning for autonomous bulldozers: Comparison with manual operation," *Int. J. Innov. Comput., Inf. Control*, vol. 15, no. 3, pp. 825–844, 2019.

[23] Z. Sun, M. Nakatani, and Y. Uchimura, "Optimal routing control of a construction machine by deep reinforcement learning," in *Proc. IEEE 15th Int. Workshop Adv. Motion Control (AMC)*, Mar. 2018, pp. 187–192.

[24] S.-K. Kim, J. S. Russell, and K.-J. Koo, "Construction robot path-planning for earthwork operations," *J. Comput. Civil Eng.*, vol. 17, no. 2, pp. 97–104, Apr. 2003.

[25] S. Lee and T. M. Adams, "Spatial model for path planning of multiple mobile construction robots," *Comput.-Aided Civil Infrastruct. Eng.*, vol. 19, no. 4, pp. 231–245, Jul. 2004.

[26] A. R. Soltani and T. Fernando, "A fuzzy based multi-objective path planning of construction sites," *Autom. Construct.*, vol. 13, no. 6, pp. 717–734, Nov. 2004.

[27] E. W. Dijkstra, "A note on two problems in connexion with graphs," *Numer. Math.*, vol. 1, no. 1, pp. 269–271, Dec. 1959.

[28] P. E. Hart, N. J. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *IEEE Trans. Syst. Sci. Cybern.*, vol. SSC-4, no. 2, pp. 100–107, Jul. 1968.

[29] J. H. Holland, *Adaptation in Natural and Artificial Systems: An Introductory Analysis With Applications to Biology, Control, and Artificial Intelligence*. Cambridge, MA, USA: MIT Press, 1992.

[30] T. Cheng, U. Mantripragada, J. Teizer, and P. A. Vela, "Automated trajectory and path planning analysis based on ultra wideband data," *J. Comput. Civil Eng.*, vol. 26, no. 2, pp. 151–160, Mar. 2012.

[31] G. Bohács, A. Gyimesi, and Z. Rózsa, "Development of an intelligent path planning method for materials handling machinery at construction sites," *Periodica Polytechnica Transp. Eng.*, vol. 44, no. 1, pp. 13–22, 2016.

[32] M. Štefanič and V. Stankovski, "A review of technologies and applications for smart construction," *Proc. Inst. Civil Eng. Civil Eng.*, vol. 172, no. 2, pp. 83–87, May 2019.

[33] C. J. Turner, J. Oyekan, L. Stergioulas, and D. Griffin, "Utilizing industry 4.0 on the construction site: Challenges and opportunities," *IEEE Trans. Ind. Informat.*, vol. 17, no. 2, pp. 746–756, Feb. 2021.

[34] Y. Xiang, H. Wang, T. Su, R. Li, C. Brach, S. S. Mao, and M. Geimer, "KIT MOMA: A mobile machines dataset," 2020, *arXiv:2007.04198*. [Online]. Available: http://arxiv.org/abs/2007.04198

[35] Y. Xiang and M. Geimer, "Optimization of operation strategy for primary torque based hydrostatic drivetrain using artificial intelligence," in *Proc. 12th Int. Fluid Power Conf. Fluid Power Netw.*, 2020, pp. 1–9.

[36] Y. Xiang, T. Tang, T. Su, C. Brach, L. Liu, S. Mao, and M. Geimer, "Fast CRDNN: Towards on site training of mobile construction machines," 2020, *arXiv:2006.03169*. [Online]. Available: http://arxiv.org/abs/2006.03169

[37] I. Kurinov, G. Orzechowski, P. Hamalainen, and A. Mikkola, "Automated excavator based on reinforcement learning and multibody system dynamics," *IEEE Access*, vol. 8, pp. 213998–214006, 2020.

[38] Y. Xiang, D. Li, T. Su, Q. Zhou, C. Brach, S. S. Mao, and M. Geimer, "Where am I? SLAM for mobile machines on a smart working site," 2020, *arXiv:2011.01830*. [Online]. Available: http://arxiv.org/abs/2011.01830

[39] Q. Zhou, J. Li, B. Shuai, H. Williams, Y. He, Z. Li, H. Xu, and F. Yan, "Multi-step reinforcement learning for model-free predictive energy management of an electrified off-highway vehicle," *Appl. Energy*, vol. 255, Dec. 2019, Art. no. 113755.

[40] Q. Zhou, Y. Zhang, Z. Li, J. Li, H. Xu, and O. Olatunbosun, "Cyber-physical energy-saving control for hybrid aircraft-towing tractor based on online swarm intelligent programming," *IEEE Trans. Ind. Informat.*, vol. 14, no. 9, pp. 4149–4158, Dec. 2017.

[41] R. Stern, "Multi-agent path finding—A overview," in *Artificial Intelligence, RAAI Summer School 2019*. Cham, Switzerland: Springer, 2019, pp. 96–115.

[42] N. R. Sturtevant, "Benchmarks for grid-based pathfinding," *IEEE Trans. Comput. Intell. AI Games*, vol. 4, no. 2, pp. 144–148, Jun. 2012.

[43] J. Švancara, M. Vlk, R. Stern, D. Atzmon, and R. Barták, "Online multi-agent pathfinding," in *Proc. AAAI Conf. Artif. Intell.*, 2019, vol. 33, pp. 7732–7739.

[44] A. Madkour, W. G. Aref, F. U. Rehman, M. A. Rahman, and S. Basalamah, "A survey of shortest-path algorithms," 2017, *arXiv:1705.02044*. [Online]. Available: http://arxiv.org/abs/1705.02044

[45] A. Felner, R. Stern, S. E. Shimony, E. Boyarski, M. Goldenerg, G. Sharon, N. Sturtevant, G. Wagner, and P. Surynek, "Search-based optimal solvers for the multi-agent pathfinding problem: Summary and challenges," in *Proc. SoCS*, 2017, pp. 29–37.

[46] E. Boyarski, A. Felner, G. Sharon, and R. Stern, "Don't split, try to work it out: Bypassing conflicts in multi-agent pathfinding," in *Proc. ICAPS*, 2015, pp. 47–51.

[47] G. Sharon, R. Stern, A. Felner, and N. Sturtevant, "Meta-agent conflict-based search for optimal multi-agent path finding," in *Proc. SoCS*, vol. 1, 2012, pp. 97–104.

[48] A. Felner, J. Li, E. Boyarski, H. Ma, L. Cohen, T. K. S. Kumar, and S. Koenig, "Adding heuristics to conflict-based search for multi-agent path finding," in *Proc. ICAPS*, 2018, pp. 83–87.

[49] J. Li, A. Felner, E. Boyarski, H. Ma, and S. Koenig, "Improved heuristics for multi-agent path finding with conflict-based search," in *Proc. 28th Int. Joint Conf. Artif. Intell.*, Aug. 2019, pp. 442–449.

[50] J. Li, D. Harabor, P. J. Stuckey, A. Felner, H. Ma, and S. Koenig, "Disjoint splitting for multi-agent path finding with conflict-based search," in *Proc. ICAPS*, vol. 29, 2019, pp. 279–283.

[51] G. Gange, D. Harabor, and P. J. Stuckey, "Lazy CBS: Implicit conflict-based search using lazy clause generation," in *Proc. ICAPS*, vol. 29, 2019, pp. 155–162.

[52] W. Hönig, S. Kiesel, A. Tinka, J. W. Durham, and N. Ayanian, "Conflict-based search with optimal task assignment," in *Proc. AAMAS*, 2018, pp. 757–765.

[53] M. Barer, G. Sharon, R. Stern, and A. Felner, "Suboptimal variants of the conflict-based search algorithm for the multi-agent pathfinding problem," in *Proc. SoCS*, 2014, pp. 19–27.

[54] P. Fankhauser and M. Hutter, "A universal grid map library: Implementation and use case for rough terrain navigation," in *Robot Operating System (ROS)*, vol. 625. Cham, Switzerland: Springer, 2016, pp. 99–120.

[55] T. Ikeda, M.-Y. Hsu, H. Imai, S. Nishimura, H. Shimoura, T. Hashimoto, K. Tenmoku, and K. Mitoh, "A fast algorithm for finding better routes by AI search techniques," in *Proc. Vehicle Navigat. Inf. Syst. Conf.*, Aug. 1994, pp. 291–296.

YUSHENG XIANG (Member, IEEE) received the M.Sc. degree in vehicle engineering from Karlsruhe Institute of Technology, Karlsruhe, Germany, in 2017, with a focus on mathematical model building and simulation, where he is currently pursuing the Ph.D. degree with the Institute of Vehicle System Technology. He was a Research Scientist with Robert Bosch GmbH, Germany. Since September 2020, he has been a Visiting Scholar with the University of California at Berkeley, USA, supervised by Prof. Samuel S. Mao. He is also the CTO of Elephant Tech LLC, Shenzhen, China, which is a spinoff from Prof. Samuel S. Mao's Laboratory. He has authored nine influential journals and international conference papers, and holds five patents. His group deals with the improvement of mobile machines' performance using artificial intelligence and the Internet of Things.

KAILUN LIU (Student Member, IEEE) received the B.Sc. degree in mechanical engineering from East China University of Science and Technology, Shanghai, China, in 2015, with a focus on process equipment and control engineering, and the M.Sc. degree in mechanical engineering from Karlsruhe Institute of Technology, Karlsruhe, Germany, in 2021, with a focus on vehicle technology. Before he has joined Prof. Geimer's Laboratory to do his master's thesis under the supervision from Yusheng Xiang, he did an internship on electric vehicles at Bosch China Central Research Institute, in 2019.

TIANQING SU (Member, IEEE) received the B.Sc. degree in telecommunication from Xidian University, Xi'an, China, in 2013, and the M.Sc. degree in information systems engineering from the Technische Universität Braunschweig, Germany, in 2017. She is currently pursuing the Master of Business Administration degree with Guanghua School of Management, Peking University, Beijing, China. Before she found Elephant Tech LLC, Shenzhen, and become the first CEO of this hightech startup, she was a Software Engineer in hybrid and electric vehicle with Continental AG, Regensburg, Germany, and Robert Bosch GmbH, Abstatt, Germany.

JUN LI (Member, IEEE) received the B.S. degree in computer science and technology from Shandong University, China, in 2003, the M.Sc. degree in information and signal processing from Peking University, Beijing, China, in 2006, and the Ph.D. degree in computer science from Queen Mary University of London, U.K., in 2009. He is currently a Senior Lecturer with the Artificial Intelligence Institute, and the Faculty of Engineering and Information Technology, School of Computer Science, University of Technology Sydney, Australia. He also leads the AI Department, Elephant Tech LLC. His research interests include probabilistic data models and image and video analysis using neural networks.

SHIRUI OUYANG received the B.Sc. degree in vehicle engineering from Beijing Institute of Technology, Beijing, China, and the M.Sc. degree in vehicle engineering from Karlsruhe Institute of Technology, Karlsruhe, Germany, in 2017, with a focus on general vehicular technology and mobile machines, where he is currently pursuing the Ph.D. degree with the Institute of Vehicle System Technology. His research interests include improving the efficiency of mobile working machines by implementation of the hydraulic energy recovery systems and the optimization its control strategy.

SAMUEL S. MAO received the Ph.D. degree from the University of California at Berkeley, in 2000. After that, he started his career at Lawrence Berkeley National Laboratory, where he was a Career Staff Scientist, until 2013. He returned to U.C. Berkeley campus, as an Adjunct Professor, in 2004, when he also established the Clean Energy Engineering Laboratory that has spun off an international technology development and commercialization institution, and the Institute of New Energy, launched in 2013. Having published 180 refereed research articles that have received more than 46,000 citations, he is currently an Inventor of 80 patents in the U.S. and abroad. He delivered nearly 100 invited, keynote or plenary speeches at international conferences, and previously served as a Technical Committee Member, a Program Review Panelist, a Grant Proposal Evaluator, and a National Laboratory Observer for the U.S. Department of Energy. He received an "R&D 100" Technology Award for his technological innovation, in 2011, and Berkeley MEGSCO Faculty Teaching Award for his dedication to higher education, in 2008. In addition to co-founding three international materials and energy technology conferences, he co-chaired the Materials Research Society (MRS) annual meeting, in Spring 2011, and the International Conference on Clean Energy, in 2012.

MARCUS GEIMER received the Diploma degree in mechanical engineering from RWTH Aachen University, Aachen, Germany, in 1990, and the Ph.D. degree from the Institute of Hydraulics and Pneumatics (currently, the Institute for Fluid Power Drives and Systems), RWTH Aachen University, in 1995. In 1995, he started his industrial career in the field of construction, where he was the Leader of the research group for Hydraulic Breakers. In 2000, he changed to the hydraulic industry, where he leads the construction and customer development for mobile hydraulics. Since 2005, he has been a Full Professor and the Director with the Institute of Mobile Machines (Mobima), Karlsruhe Institute of Technology (KIT), Germany. His research interests include drives and controls for mobile working machines, such as agriculture, construction, and municipal vehicles. Research projects on hydrostatic, electric and hybrid drives, and traction as on function drives, have been successfully completed. Modern control strategies, such as machine learning methods, neural networks, or predictive control, are under research.

• • •