Automation in Construction 155 (2023) 105071

Contents lists available at ScienceDirect

Automation in Construction

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

Automatic high-level motion sequencing methods for enabling multi-tasking construction robots

Xi Wang [a], Shuoqi Wang [b], Carol C. Menassa [c,*], Vineet R. Kamat [c], Wes McGee [d]

[a] Department of Construction Science, Texas A&M University, 3137 TAMU, College Station, TX 77843, USA
[b] Department of Robotics, University of Michigan, 2505 Hayward Street, Ann Arbor, MI 48109, USA
[c] Department of Civil and Environmental Engineering, University of Michigan, 2350 Hayward Street, Ann Arbor, MI 48109, USA
[d] Taubman College of Architecture and Urban Planning, University of Michigan, 2000 Bonisteel Boulevard, Ann Arbor, MI 48109, USA

ARTICLE INFO

Keywords:
Human-robot collaboration
Learning from demonstration
Construction robot
Digital twin
Building information modeling
Robot sequential motions

ABSTRACT

Robots are expected to play an important role in future construction work. However, they are not yet widely adopted by the industry because it is difficult and expensive to program robots to conduct a variety of construction tasks. This paper presents a method for intuitively and flexibly teaching robots to perform various construction tasks through demonstrations. Robots are first programmed with basic skill primitives and then learn the sequencing of these primitive skills to perform different types of construction work under the guidance of human supervisors. The construction workflow and the interaction processes are enabled by a process-level digital twin system. Case studies with three assembly scenarios and a wooden frame construction experiment are used to present and verify the proposed method. The proposed approach enables automatic robot motion sequencing abilities through Learning from Demonstration and has the potential to enable the widespread adoption of robots on construction sites.

1. Introduction

The construction industry plays an important role in modern civilization and accounts for 13% of the global Gross Domestic Product (GDP) [1]. Despite its significant role in the economy, the industry is facing several issues that affect its long-term development. On one hand, the construction industry has stagnant productivity, which not only extends the duration of construction projects themselves but also delays other activities that will take place in the constructed facilities [2,3]. On the other hand, the industry is facing a serious shortage in its workforce, with 73% of general contractors in the U.S. reporting the issue in 2022 [4,5]. The heavy physical demand poses a high risk of work-related musculoskeletal disorders to the workers and excludes many female workers and people with physical disabilities from the construction workforce, which further aggravates the issue [6].

Robots have been widely proposed as a solution to reduce the physical workload of construction workers and mitigate productivity and labor shortage issues in the construction industry [2,7,8]. Several commercial construction robots have been developed and entered the market in recent years, such as the layout drawing robot "Dusty" by Dusty Robotics [9], the bricklaying robot "SAM" by Construction Robotics [10], the rebar tying robot "Tybot" by Advanced Construction Robotics [11], and the overhead drilling robot "Jaibot" by Hilti [12]. However, many construction robots are single-task robots that are specifically designed to conduct one type of construction work [13]. These robots can only be used at a certain stage of a construction project [14]. Meanwhile, the construction industry is fragmented [15]. Many contractors have a limited number of projects going on at the same time and the projects can have long physical distances between each other. It is not cost-effective for these contractors to invest in robots that can only perform a certain type of work and stay inactive during other phases of the project. As a result, construction robots are not widely adopted by the industry nowadays.

Therefore, multi-task robots that can be configured to perform different types of construction work and be involved in multiple construction stages are needed. However, a construction project includes many different types of construction tasks. It is a significant effort to preprogram robots with a wide variety of construction tasks, especially when robotic engineers who program the robot lack construction domain knowledge [16]. Because of the complexity and uncertainty of construction work, improvisation, and adjustments are frequently required to perform the work, which adds to the difficulty of

* Corresponding author.
E-mail address: menassa@umich.edu (C.C. Menassa).

https://doi.org/10.1016/j.autcon.2023.105071
Received 1 May 2023; Received in revised form 20 August 2023; Accepted 22 August 2023
Available online 2 September 2023

preprogramming robots to effectively perform a great number of construction tasks [17,18]. Moreover, with preprogramming, the robot follows the previously determined motion sequence until all the motions in the sequence are completed. As a result, the robot cannot adapt to failures at the intermediate steps (e.g., failure to grasp the workpiece).

When human apprentices learn construction work, they start with some basic construction skills and then they learn how to apply and sequence these skills to achieve construction task objectives. For example, apprentice carpenters usually learn some basic woodwork skills, such as cutting, nailing, filing, etc. Then, they choose and apply these skills in different sequences to perform construction work, which usually requires experience and flexibility. While there are many different types of construction tasks, the number of basic skills that need to be learned is limited. Inspired by this workflow, this paper proposes to program robots with basic construction skills (e.g., reaching, opening gripper, closing gripper, nailing) and deploy the robot to the construction site to learn how to sequence these skills to flexibly perform different construction tasks under the guidance of construction workers.

In order to enable this process, a Human-Robot Collaborative Construction (HRCC) method that integrates interactive Learning from Demonstration (LfD) with a process-level digital twin system is proposed. This method focuses only on learning the sequencing of different known motions through LfD instead of learning the execution of individual motions. More specifically, it focuses on construction assembly tasks by enabling a consolidated workflow from preprocessing assembly parts, workpiece manipulation, and connecting the assembly parts, to performing finishing work, which are the most common tasks in construction. The proposed method is based on robotic manipulators, which offer high degrees of freedom and have high flexibility to be configured for a variety of complex construction tasks [19]. The robot is equipped with the proposed system and a set of parameterized skill primitives when delivered to construction sites. The system allows robots to automatically sequence their motions by interactively requesting and learning from human demonstrations. While the robot mostly follows human co-workers' guidance as a novice in the beginning, it becomes increasingly intelligent and independent as it learns more from its interaction with humans.

A process-level digital twin system adapted from the authors' previous work is used for construction site monitoring, computing, and bi-directional interaction between the human and the robot [20]. It integrates the Robot Operation Environment (ROE), a Building Information Model (BIM), the Robot Operating System (ROS), and a Graphical User Interface (GUI). The system takes information from the BIM for task planning. It senses the state of the construction site and uses the state together with the task objectives to decide its next action or request a human demonstration. During the construction process, human workers and robots collaborate as master-apprentice teams. Human workers take the role of supervisors who teach, supervise, and intervene in the robotic construction process. Robots act as apprentices and are responsible for task planning, motion planning, and task execution. A Scene Distance Array (SDA) based on multi-layer scene state representation arrays has been proposed. SDAs serve as the identifier to record and retrieve knowledge. During the learning stage, robots build and save probabilistic mappings from the SDA to modular skill primitives in their knowledge base based on human demonstrations in the digital twin system. The learned knowledge, which can be shared among robots, is then used for automatic high-level motion sequencing for different types of construction tasks. During the decision-making stage, the robot queries the knowledge base with the SDA to find the corresponding primitive to execute. Based on the proposed approach, a delivery framework for construction robots is presented.

The proposed method is demonstrated through case studies that contain three types of construction tasks, exterior wall sheathing, drywall installation, and timber frame construction. A wooden frame construction experiment in Gazebo simulation is used to verify the proposed system as a proof-of-concept implementation. The proposed construction robot delivery framework, along with the robot task learning and sequential motion planning capabilities, has the potential to improve the usage and cost efficiency of construction robots, and thus improve the acceptance and deployment of HRCC.

This paper is organized to first present a detailed review of existing methods of robot LfD and how to determine the motion sequences of robots (Section 2). Related research gaps are discussed. Next, skill primitives for construction robots and the technical approach to enable automatic high-level motion sequencing of construction assembly tasks are discussed in Section 3. Then, case studies and the proof-of-concept implementation are presented in Sections 4 and 5. Finally, the case study and implementation results are analyzed and discussed in Section 6, followed by the conclusions of this research (Section 7).

2. Related work

2.1. Robot learning from demonstration

LfD enables robots to acquire skills by observing and imitating human actions or decisions [21]. It does not require robot programming expertise from the teachers or demonstrators, and thus allows experts with greater domain knowledge, such as construction workers, to teach the robot to perform work in the specific domain (e.g., construction) [22]. A typical LfD problem often entails creating a mapping between the environment state and the robot's action. However, since robots are usually unable to fully observe the environment state, most LfD studies also incorporate an observed state, which serves as a bridge between the environment state and the robot's action. In this situation, both the mapping from the environment state to the observed state and the mapping from the observed state to the robot action need to be considered in the framework design.

Robots can learn skills on different levels through LfD. Low-level LfD focuses on learning motor policy for manipulation, referred to as "trajectory encoding", such as learning peg-in-hole or slide-in-groove [23–25]. Compared to computational-based motion planners, motor control policy learned from demonstrations has higher adaptability and can solve more complex cases. On the other hand, high-level LfD learns "symbolic encoding" about how to organize predefined motion primitives into a sequence, such as sorting colored blocks into different bowls [25,26]. With the goal of achieving automatic sequencing of high-level motions for construction assembly, this paper focuses on high-level LfD.

There are several demonstration methods for high-level LfD. Humans can demonstrate motion sequences using the kinesthetic approach by physically moving the robot's passive joints [27]. While allowing demonstrations to be identically transferred to the robot, this approach is restricted by the scale of the task and safety considerations. Thus, teleoperation is used instead, such that the human movements are captured by vision-based systems [28], motion capture systems [29], hand-coded controllers [30], Virtual Reality (VR) [31,32], or a combination of several sources [33–35], and used as demonstrations. Nevertheless, these demonstrations are continuous movements, which need to be segmented into motion primitives for the robot to learn the sequencing of primitives [36]. Some demonstration approaches can directly use motion primitives to achieve high-level task objectives thereby avoiding the segmentation process. The most widely-used method is to use the GUI to select the primitive [37,38]. Additionally, language-based demonstrations can guide the robot to implement certain motion primitives in specific sequences with language instructions [39,40].

2.2. Robots' motion sequence determination

Although advancements in machine learning have greatly improved robot intelligence in recent years, direct programming is still the most common method to determine motion sequences for construction robotic manipulators [41–43]. Another popular approach is to guide the robot through a series of motions continuously through teleoperation or step-by-step to complete construction tasks [44,45]. Some studies have used automatic planners such as the Stanford Research Institute Problem Solver (STRIPS) to determine robot motion sequences [46,47]. However, it is extremely challenging to accurately and consistently account for all pre-conditions and post-conditions when defining construction movements for these planners.

Hierarchical task modeling organizes different levels of task primitives of a complex task into a hierarchical structure and uses symbolic representations to identify task primitive relations [48]. The task primitives and their relationships in the hierarchical model greatly reduce search space and thus improve robot planning efficiency. It has been used to represent the knowledge the robot learned from human demonstrations [37,38]. However, the learned motion sequence has limited flexibility and adaptability. Hierarchical task networks and And-Or-Graph have been used to represent knowledge achieved from human demonstrations to facilitate robot planning on the sequencing of motion primitives [49,50]. These studies mainly focus on resolving multiple sequential solutions of individual tasks and emphasize the combination of different demonstrated sequences to provide the most preferred option.

Reinforcement learning is also a popular approach for robot sequential decisions in its motions [51]. It allows the robot to conduct exploration by itself and provides feedback to the robot with rewards to guide the robot to find the solution [52]. In [53,54], the authors used two separate policy models to decide which motion primitive to select and the parameters of the selected motion. The policy is trained in simulation and is then transferred to the physical world for execution. In order to further reduce the robot exploration load and accelerate the reinforcement learning process, some studies take advantage of the robot's prior experience [55,56]. Demonstration is also a popular approach to be used together with reinforcement learning [16,51]. Explorations are biased toward demonstrated actions to expedite the trial-and-error process [57,58].

2.3. Summary

The aforementioned approaches present certain limitations in generating motion sequences for construction robots. First, construction work typically occurs on a large scale and involves manipulation of large and heavy construction components. Therefore, it takes considerable effort and consumes significant resources for human workers to demonstrate one type of task repeatedly in the physical world. It also makes demonstrations through direct robot operations like motion capture or kinesthetic approaches infeasible. Moreover, effectively mapping the environment state to the observed state presents a significant challenge. While most approaches use vision-based systems to model scene states [39,59], identifying object relationships and detecting scene changes in a construction environment directly from the camera is difficult due to the large space and complexity of the environment.

Second, construction assembly encompasses a wide variety of tasks, making it crucial to enable the previously acquired motion sequencing ability to be transferred to other tasks with minimal additional demonstrations required. While some existing studies transferred previously learned skills to different tasks, the task differences tend to be limited, such as picking up a tray versus picking up a stud [53,54].

Third, construction assembly tasks are typically quasi-repetitive, requiring workpieces to be picked up from and installed in different locations. This necessitates the parametrization of motions (i.e., taking inputs from the environment), which increases the difficulties to explore solutions through methods like reinforcement learning. It is also challenging to accurately segment construction operations from a movement sequence detected by motion sensors or cameras.

Lastly, construction involves numerous uncertainties. It also involves a variety of operations that are extremely challenging for the robot to solve through exploration or automatic planners. Rather than adhering to a fixed predetermined or learned sequence, robots must adapt their plan based on work status to effectively execute construction tasks. Thus, the system should also allow human supervision (e.g., previewing the robot plan before execution) and intervention (e.g., modifying the robot plan) to ensure construction safety and quality.

An efficient HRCC system should enable construction robots to develop motion sequences that are robust to uncertainties, generalizable across different types of construction assembly tasks, supports parameterized motion primitives, and only requires a minimal number of prior exploration trials or human demonstrations. To address these issues, this paper adapts the previously proposed closed-loop BIM-driven HRCC digital twin system [20] and incorporates the LfD module into the workflow. The digital twin enables real-time monitoring of the robotic construction workspace, providing effective construction workspace scene state observations. Additionally, it provides a 3D/VR interface that allows human co-workers to observe the scene states and facilitates intuitive human demonstrations through direct selection of primitives from the interface.

3. Technical approach

3.1. Digital twin system for BIM-driven HRCC

In this system, the relationship between the human co-worker and the robot is based on the master-apprentice model. The human co-worker is the master and supervisor, and the robot is the apprentice and learner. At the onset, the robot has no knowledge about sequencing its skills to perform construction tasks. Thus, the human co-worker needs to show the robot how to do the work through demonstrations. After demonstrations, the robot learns and remembers the knowledge. When they perform the same or similar tasks in the future, they can apply what they learned to automatically perform the task on their own. To ensure the safety and reliability of the system, we propose to have the human co-worker supervise the process and approve each step before the robot actually takes any physical actions. The HRCC process is achieved through a closed-loop digital twin system for BIM-Driven HRCC adapted from the authors' previous work [20]. The proposed LfD process is integrated into the system. Fig. 1 shows the system framework. The following paragraphs provide a brief overview of the system and highlight the differences compared to the previous version, whereas more details of the system can be found in [20].

The digital twin system consists of three main modules. The ROE represents the construction site, robots, and sensors. Since this study mainly focuses on the LfD algorithms, the ROE is simulated in the Gazebo robotics simulator. However, the system has the capability to work with physical robots through fiducial marker-based object localization and TwinCAT Automated Device Specification (ADS), which has been discussed in detail in [20,60].

The user interface is implemented in Unity with both 3D and VR options. It reflects construction site status in real-time and provides augmented information (e.g., the as-designed scene from the BIM) to assist with human co-workers' decision-making. The automatically generated digital twin also allows human co-workers to teach and send instructions to robots, supervise the construction work process, and intervene in the robot's work when necessary. In addition to adding interactive functions to the previous system to enable LfD, the current system queries workpiece poses multiple times during the construction process. The previous system only detects material stacking poses at the beginning of the task and follows the predefined motion sequence to conduct the pick-and-place operation. Timely reflection of workpiece poses to human co-workers and the robot through the interface can make them aware of interruptions and intermediate failures, which are critical to deciding the next-step motion in the sequence.

As the middleware of the system, ROS enables communication among the user interface, robot, sensors, and the BIM on different

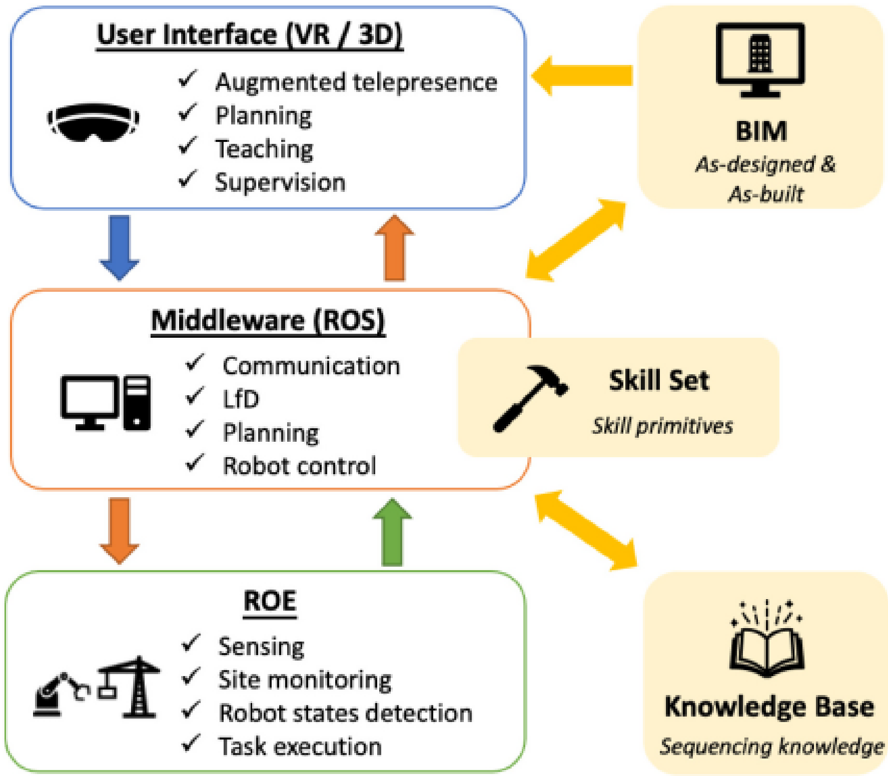

Fig. 1. Digital twin system framework.

devices through the local area network. It is also the central processing unit of the system that controls the workflow, fuses and processes sensor data, and performs most of the computation work. ROS is also the platform to create the skill set of robots. Skills of the robot are implemented in ROS programs and are activated to plan the trajectories and control the robot during the teaching and construction processes. ROS can bi-directionally communicate with the BIM and the knowledge base to retrieve and save construction data and acquired knowledge.

For the BIM to support the robotic construction process enabled by the proposed digital twin system, the BIM needs to contain (1) shop-drawing level geometry; (2) construction sequence data; (3) workpiece relationships (i.e., ontology); (4) object layers based on a predefined layer structure (whether the object belongs to Targets, Materials, As-Built, As-Designed, or Virtual Collision); and (5) necessary robot operation support (e.g., component attribute data such as how to grip the component). A detailed discussion can be found in Section 3.2 of [20]. It must be noted that some of these categories of information in the BIM can be or have high potential to be automatically generated in the future, although they are manually created in this research [61,62].

3.2. Skill primitives for construction robots

Previous research has utilized motion primitives, defined as robot movement generators that can be reused in a modular manner, for the purposes of robot programming and learning [28,63,64]. In this research, we build upon this concept and extend it into skill primitives for construction robotics applications. Skill primitives can be defined as previously acquired modular skills that robots can utilize directly to perform a given task. They can be grouped into three categories: motion primitives, sensing primitives, and reasoning primitives.

The motion primitives serve as the fundamental operation skills for successfully carrying out construction work. Previous research has explored the idea of construction operation modularization. For instance, [65] identified ten basic activities, including placing, connecting, attaching, finishing, coating, concrete building, inlaying, covering, and jointing for robotics building construction. In another study, eleven basic tasks (i.e., connect, cover, dig, inspect, measure, place, plan, position, spray, spread, finish) are proposed as a construction automation taxonomy, which can be conducted by both construction workers and equipment [66]. Building on this taxonomy, [67] proposed a construction automation methodology by adding information input essential for each type of basic task.

Despite the efforts in previous studies, the proposed elements fail to provide the requisite details necessary to program robot motions. Taking the "connect" basic task as an example, which is defined as "join or fasten together" (e.g., nail). However, programming a robot to nail requires a sequence of two to three motions, including positioning the robot end-effector in the correct location, using it to drive the nail, and potentially withdrawing the robot end-effector from the workpiece. Although [66] suggests using elemental motions to break down basic tasks, examples for construction robots or machines are not given. Therefore, construction motions that are modularized and reusable by the robot as primitives are needed.

In order to facilitate robot programming and motion sequencing, we propose to treat motion primitives as a dynamic and flexible repository adapting "Basic Tasks" and "Elemental Motions" from [66]. For example, the basic task of "place" is disassembled into primitives of "reach" and "open gripper", which can be reused in other tasks such as "pick" ("reach" and "close gripper"). Some tasks like "spray" involve two simultaneous elemental motions of end-effector moving along a trajectory while spraying, which are coded as one motion primitive. New primitives can be dynamically added to the collection. Taking ceiling tile placement as an example, it is challenging to automatically generate the motion plan to manipulate the tile through the ceiling grid with motion planners. However, robots can learn the manipulation processes from construction workers' demonstrations [17]. In this case, the process is considered a standalone motion primitive even though it is a "reach" motion in reality.

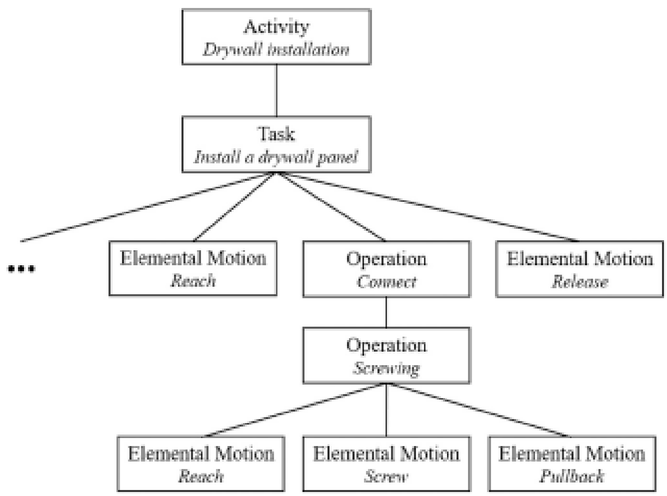

Fig. 2. Example of proposed hierarchical taxonomy.

In addition to motion primitives that directly result in construction progress, sensing and reasoning primitives are also essential skills for robotic construction. Sensing primitives refer to skills that enable a robot to comprehend its environment by perceiving various sensory inputs, including visual, auditory, and haptic feedback (e.g., speech recognition and mapping). Reasoning primitives, on the other hand, serve as the robot's brain. Reasoning primitives take the form of system function modules of learning, processing, and computation, such as solvers or trained models.

In previous studies, a unified hierarchy to describe construction work is missing, making it challenging to develop a coherent framework to modularly program and teach construction robots. To overcome this ambiguity, this study adopts a bottom-up hierarchical taxonomy—elemental motion, operation, task, and activity—by adapting concepts from prior studies [65,66,68]. Elemental motions are reusable motion primitives (e.g., reaching, closing gripper, spraying along a path). An operation consists of a sequence of elemental motions and/or other operations. For example, the nailing operation requires the robot end-effector to move to the nailing location, activate the nail gun, and pull the end-effector away from the workpiece. A construction task refers to a completed piece of work, such as installation of a piece of curtain panel. An activity represents a subdivision of a construction project as the unit in the work breakdown structure, such as "Install drywall" [69]. Fig. 2 shows an example of the proposed hierarchical taxonomy of a drywall installation activity. This research aims at sequencing elemental motions to perform construction activities, which can be broken down into construction tasks by querying task targets one by one from the sequence.

3.3. LfD module overview

With the concept of skill primitives, we conceive a framework for construction robot delivery. Robotic engineers apply the proposed digital twin system with the LfD module to the robot. Then, they program the robot with a set of reasoning, motion, and sensing primitives. After the robot arrives at the construction site, it learns how to sequence these primitives and, if needed, new primitives, to perform different types of construction tasks. This delivery framework has two key strengths. Firstly, the knowledge gap between two domains, construction and robotics, is bridged. Robotic engineers apply their programming expertise to software development, and construction workers provide high-level instructions for specific construction work, which improves both development productivity and quality of the produced work. Secondly, the robot can conduct multiple construction tasks and constantly acquires new knowledge by learning from workers. As a result, it is more cost-efficient compared to single-purpose robots that are active at certain stages of a construction project and remain idle for an extended period.

The paper presents the integration of the LfD module with the digital twin system. Then, it demonstrates how construction workers use the system to teach robotic construction assembly work, to enable automatic sequencing decisions, and to supervise the process. It needs to be pointed out that a construction assembly task typically entails multiple tools, and thus requires a system that incorporates various robots or devices. This study assumes that primitives have been previously distributed to distinct robots with different end-effector tools or devices such as Computer Numerical Control (CNC) machines. The device or end-effector tool that is accountable for the specific motion primitive will be used for planning and execution at each step. To ensure clarity and brevity, the term "robot" denotes the automation system that consists of several robots and viable devices in the subsequent sections.

In the course of interactive LfD, robots acquire knowledge by actively seeking and processing demonstrations [70]. This interaction process is facilitated by the digital twin system, which enables human-robot bi-directional communication, robot control and supervision, and BIM data extraction. By actively tracking object locations and updating components relationships (e.g., workpiece grabbed or not grabbed by the robot), the digital twin also enables mapping from the environment to the observed scene state. When the robot is delivered to the construction site, skill primitives programmed by robotic engineers are stored as reusable components, which are accessible by the human co-worker from the GUI of the digital twin. These primitives need to be triggered by the robot as learned knowledge or by the human co-worker as demonstrations. The system also comes equipped with some sensing and reasoning functions that activate under specific conditions [20]. For example, the digital twin needs to be generated by loading the BIM and construction workspace data when construction starts [20]. The robot needs the capabilities to detect and localize workpieces and to model the workspace for collision avoidance [71–74].

The learning outcome is that the robot is capable of autonomously performing construction assembly activities by selecting primitives in a step-by-step manner under human supervision. One task is addressed at a time. The target workpiece is decided either by the construction sequences in the BIM or by the human co-worker. The acquired knowledge applies to subsequent targets in the current and future construction tasks and can be shared with other robots.

Fig. 3 depicts the LfD workflow. At the onset of a construction activity, the target workpiece suggested by the BIM is highlighted in the GUI. Upon human approval or reselection, an initial SDA is created. The robot then searches its knowledge base to identify the SDA (Section 3.4). If the robot is unable to recognize the SDA, it will request its human co-worker to demonstrate the primitive. After observing the scene from the digital twin, the human co-worker can intuitively make demonstrations by selecting from the drop-down lists in the GUI. The mapping from the SDA to the demonstration is saved in the robot knowledge base to facilitate automated invocation of this knowledge later, without necessitating a demonstration when the SDA is encountered again. When the SDA exists in the knowledge base, the corresponding skill primitive of the SDA is selected using the probabilistic mapping saved in the robot knowledge base.

In order to reduce the risk of hazardous consequences caused by incorrect or suboptimal robot decisions, primitives selected by the robot need to go through human co-workers' confirmation through the interface in the digital twin. To ensure consistent safety, robot actions in the current workflow will not be executed without human approval. The selected primitive is shown to the human co-worker on the user interface. The human co-worker can either approve the primitive or opt to choose another primitive as a demonstration, and the robot will learn the knowledge. As a result, the system can always prioritize the human co-worker's preference.

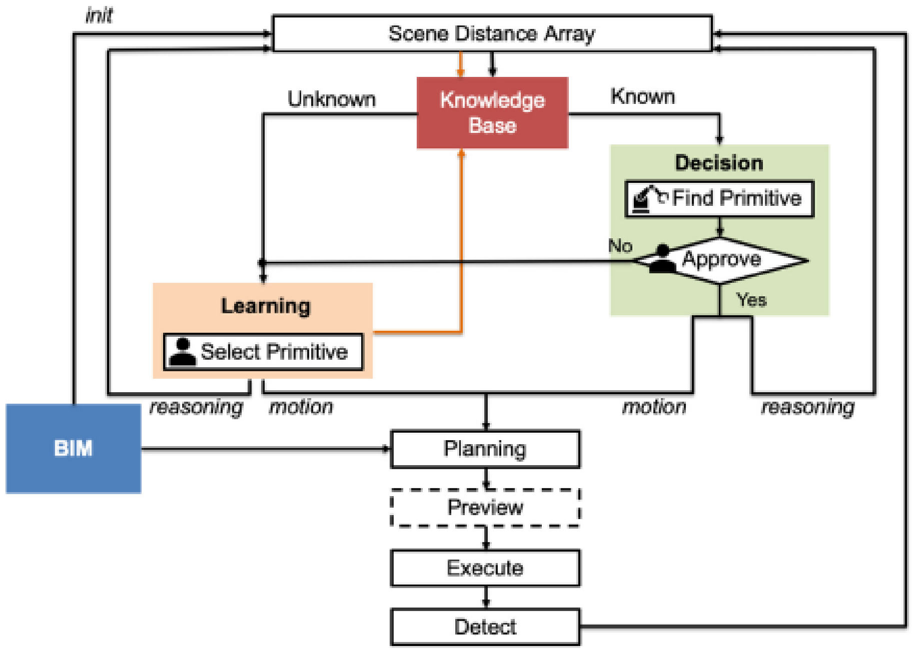

Fig. 3. LfD workflow.

If a reasoning primitive is determined, the system will execute the corresponding function and update SDA accordingly. If the approved primitive is a motion primitive, the robot will acquire input parameters (e.g., 6DOF reaching poses) needed for motion planning (if applicable). The parameter can be obtained from BIM, human demonstration, or a combination of both. For robot manipulation (e.g., reaching the material, moving the material to the target place), after the motion plan is generated, it is sent to the digital twin and processed into high-fidelity simulations of the execution process. It allows human co-workers to preview the motion plan before approving the plan for execution. If human co-workers feel the plan is not optimal, they can request the robot to generate another plan for approval. The detailed methods of motion planning, optional plan preview, and execution are outlined in [44]. Following motion completion, the SDA is updated, with the process continuing until all task-related procedures are concluded.

Instead of defining the full assembly sequence beforehand, either the robot or the human co-worker determines the subsequent primitive at each step according to the latest SDA. As a result, the system has the robustness of withstanding disruptions during the construction process, such as failure grabbing objects. In such cases, the SDA will remain the same and the robot will attempt again with the updated input parameters (e.g., the re-detected object location).

In construction assembly, there are a wide variety of tasks but the sequence of motions needed to perform an assembly task is relatively fixed. Therefore, a robot without any prior knowledge can learn and perform subsequent work only after a human co-worker has demonstrated the task, although the system also supports multiple users to provide demonstrations. The system uses two designs to handle human errors to ensure they provide effective demonstrations and guide the robot. First, the human co-workers can cancel the primitives they selected. Some primitives associated with low-level operations such as *Nail*() are irreversible after execution. However, these primitives have reversible reasoning primitives (e.g., *start_nailing*()) and sometimes come with reversible preparatory motions (e.g., reach the nailing location), which greatly reduce the possibility of mistakenly performing the *Nail*() motion. Second, the robot learned knowledge is dynamically updated. Even if the robot is initially taught to perform work in an uncommon way, the demonstrator or other demonstrators can correct it later by selecting another primitive to proceed. The probabilistic mapping in the knowledge base will update accordingly, and the robot will select the primitive with the highest possibility to proceed later.

3.4. Representing sequencing knowledge with scene distance Array

While robots and human co-workers are deciding the motions to take, two scenes are taken into account: the current scene showing the present construction environment and the target scene reflecting the desired goal. Therefore, the SDA used to determine the primitive needs to consider both the current and the target scene. Additionally, due to the nature of construction tasks, several requirements should be met when designing the data structure of SDAs. First, construction comprises a broad range of assembly tasks with various motion sequences, thus SDAs need to be flexible to depict a wide variety of construction assembly situations. Moreover, an excessive variety of SDAs will impose a considerable demonstration workload on human workers. Therefore, SDAs should be general to minimize the number of demonstrations needed for similar tasks. Furthermore, the SDA should be formulated in a manner that enables the robot to construct it automatically during the construction process, using its sensor data and information retrieved from the BIM. The SDA also needs to be comprehensible to the robot and computationally efficient.

To address the identified factors, an SDA representation calculated from two multi-layer scene state representation arrays, Current State Array (CSA) and Goal State Array (GSA), is proposed. The upper layer of the proposed arrays can be better explained in the form of 4 by 4 matrices (Table 1). Cell values in the array represent the state or relationship of the specific row and column (material (M), target place (T), robot (R), and connection (C)). For example, “At”, which is calculated by Eq. 1, reflects whether two components are close enough. The threshold

Table 1
Multi-Layer Scene State Array (Upper Layer).

	M	T	R	C
M	Preparation	At	On	
T		Finishing		
R	At	At	Withdraw	
C				Connection

Table 2
Multi-Layer Scene State Arrays (Transit Layer).

ID	Num	ID	Num
id1	Number finished	id1	Number needed
id2	Number finished	id2	Number needed
•••	•••	•••	•••
(a) Current state		(b) Goal State	

Table 4
Examples of Transit-Layer SDAs that Require Unnecessary Demonstrations.

ID	Num	ID	Num
1	2	1	2
2	6	2	4
A		B	

coefficient is a constant value determined by the tolerance of the construction task or operation. Multiple constant values can be used for different tasks or operations when the tolerances are different, such as using one value for material placement and another value for nailing.

"On" in Cell MR shows whether the material is being held by the robot. Cell MM, TT, and CC respectively represent material preparation (e.g., cutting), target finishing (e.g., painting), and connection. These values are 1 in the GSA if the corresponding operation is needed. In CSA, values in these cells represent the percentage of completed operations in the corresponding category. Moreover, Cell RR in the CSA indicates whether the robot has withdrawn its end-effector. After withdrawal, the value changes from 0 (false) to 1 (true). Cell RR in the GSA is usually set to 1, indicating the robot should withdraw its end-effector for a specific distance instead of staying in contact with the workpiece. The form of 2D array is only used for explanation purposes. In the software, the arrays are saved as vectors with only occupied cells for better efficiency.

$$At(A,B) = \begin{cases} 1, & \text{distance } (\mathrm{A}, \mathrm{B}) \leq \text{threshold} \\ 0, & \text{else} \end{cases} \tag{1}$$

SDA is calculated by deducting CSA from GSA, signifying what is still missing to achieve the goal and subsequently determining the next motion. The robot selects the corresponding skill primitives based on the SDA, which either updates SDAs (for reasoning primitives) or plans / executes motions (for motion primitives). In the cases when one SDA is projected to multiple primitives, the robot by default selects the primitive with the highest correspondence probability. Nevertheless, the system offers human co-workers the ability and the flexibility to improvise and guide the robot through the way they prefer, by giving humans the option to deny the step proposed by the robot and select another primitive to proceed. The probabilistic mapping updates each time a primitive, either proposed by the robot or selected by the human co-workers, is executed.

To minimize the demonstrations required for robots to learn various tasks, we introduce transit-layer and bottom-layer arrays. The rationale behind incorporating multiple layers in the array stems from the observation that many assembly tasks in construction share similar sequences, with only a few steps differing between them. For instance, construction assembly has several different forms of connection operations, but they typically happen at the same stage in the sequence. The robot may need to pick up and place a workpiece and connect it with nails, or it may need to pick up and place a workpiece and connect it with screws. Requiring humans to demonstrate the entire assembly sequence to the robot repetitively is a waste of their efforts. By introducing multi-layer arrays, only the percentage completed is included in the upper-layer array. When connecting operations are required, the transit-layer array is pulled out. According to the decided operation, the bottom-layer array is created to guide the connection operation. For each step, the SDA is calculated based on the CSA and GSA on the corresponding layer at that step.

The transit-layer CSA and GSA structures are shown in Table 2. This layer determines the selection of low-level operations (e.g., screwing or nailing) in the category of a high-level operation (e.g., connecting). If a connecting operation requires nailing and caulking, a 2 by 2 array will be generated on the transit layer. Each row signifies a specific low-level operation needed for a particular high-level operation with a distinct integer identifier. The number of finished and required low-level operations is recorded in the CSA and GSA respectively. On this layer, the SDA is calculated with the algorithm in Table 3 instead of direct subtraction.

Nonetheless, transit-layer SDA has a limitation that leads to unnecessary extra human demonstrations. Taking the two SDAs from different construction tasks as an example (Table 4), the material processing operations in both tasks require two types of low-level operations, cutting (id: 1) and drilling (id: 2). *A* requires two cutting operations and six drilling operations, while *B* requires two cutting operations and four drilling operations. Although the material processing operations for the two tasks are quite similar, the knowledge learned to handle *A* cannot be applied to *B* because the robot perceives them as two distinct SDAs. To address this issue, fuzzy search is employed to match similar SDAs on the transit layer so that the robot can make the best use of previously learned knowledge to select which low-level operation to proceed with. The objective of the fuzzy search is not to guarantee that the robot selects the exact correct primitive but to have the robot provide a possible close solution to the human co-worker for approval, instead of prematurely asking for human demonstrations even though a similar case has

Table 3
Algorithm for Transit-Layer SDA Calculation.

```
Input: currentTransMat A, currentTransMat B
Output: result SDA R
for each (goalMethod, goalNum) in B:
    methodFound ← False;
    for each (currentMethod, currentNum) in A:
        if currentMethod != goalMethod then continue;
        // currentMethod equals goalMethod
        numDiff = goalNum – currentNum;
        if numDiff is 0 then R.AddRow(0, 0); // all operations of this method finished
        else R.AddRow(currentMethod, numDiff); // not all operations of this method finished
        methodFound ← True;
    if not methodFound:
        R.AddRow(goalMethod, goalNum); // operations of this method haven't started
return R;
```

Table 5
Example Transit-Layer SDAs for Fuzzy Search.

ID	Num	ID	Num
L_1	n_{L1}	N_1	n_{N1}
L_2	n_{L2}	N_2	n_{N2}
$SDA_{Learned}$		SDA_{New}	

already been learned. The human co-worker needs to approve the primitive proposed by the robot through fuzzy search. If human co-workers do not agree, they can opt to select another primitive to proceed.

The robot needs to go through each transit-level SDA in the knowledge base. Table 5 shows the two example SDAs considered at the same time. $SDA_{Learned}$ has already been learned by the robot, and the mapping knowledge of $SDA_{Learned}$ is saved in the knowledge base. It represents the situation when n_{L1} instances of Operation L_1 and n_{L2} instances of Operation L_2 are needed. SDA_{New} that the robot just encountered requires n_{N1} instances of Operation N_1 and n_{N2} instances of Operation N_2. The possible conditions and corresponding solutions are presented in Table 6.

If the two SDAs are the same ($L_1 = N_1$ *AND* $L_2 = N_2$ is True, and $n_{L1} = n_{N1}$ *AND* $n_{L2} = n_{N2}$ is True), the robot will follow the mapping of the exact matched SDA to select the primitive. If only the number of operations needed is different and the types of operations required are the same, the mapping knowledge from one SDA can be transferred to another SDA ($L_1 = N_1$ *AND* $L_2 = N_2$ is True, and $n_{L1} = n_{N1}$ *AND* $n_{L2} = n_{N2}$ is False). For example, both A and B require cutting and drilling operations (i.e., the same types of operations). The only difference is that A requires six drilling operations but B only requires four. The robot previously only learned to perform the cutting operation first in the case of A. When it comes across B, instead of requesting humans to demonstrate the selection again, the robot will match B with A and automatically select the cutting operation. Upon confirmation of the selection, mapping from B to cutting is also saved in the knowledge base.

Table 6
Fuzzy Search Conditions and Solutions.

Condition		Solution
$L_1 = N_1$ *AND* $L_2 = N_2$ is True, and $n_{L1} = n_{N1}$ *AND* $n_{L2} = n_{N2}$ is True.		Exact match. Use the primitive of $SDA_{Learned}$.
$L_1 = N_1$ *AND* $L_2 = N_2$ is True, and $n_{L1} = n_{N1}$ *AND* $n_{L2} = n_{N2}$ is False.	Only one SDA in the knowledge base can match.	Use the primitive of $SDA_{Learned}$.
	More than one SDA in the knowledge base can match, and the corresponding primitive for each matching SDA is the same.	Use the primitive of any matching $SDA_{Learned}$.
	More than one SDA in the knowledge base can match but the corresponding primitives for the matching SDAs are different.	Use the primitive of the matching SDA with the shortest Distance to SDA_{New}.
$L_1 = N_1$ *AND* $L_2 = N_2$ is False.		The two SDAs do not match. The robot needs to find other SDAs in the knowledge base that can match or request human demonstrations to proceed.

Table 7
Multi-Layer Scene State Array (Bottom Layer).

	T	R
T	Finish (motion id)	
R	At (start point)	Withdraw

For the situation where the knowledge base contains multiple SDAs with the same operation types but different corresponding primitives, the robot will choose the one with the shortest distance D (Eq. 2) by default, where $Num(A_i)$ is the number of operations with ID = i. For example, in the robot knowledge base, the A corresponds to the start of the cutting operation, but there is another SDA with one cutting operation and two drilling operations corresponding to the start of the drilling operation first, possibly due to human intervention. When the robot comes across the B, it will propose to start the cutting operation because it has a shorter distance with A.

$$D = \sum_i |Num(A_i) - Num(B_i)| \tag{2}$$

The sequence needed for a low-level operation is determined by the bottom-layer array (Table 7). For example, to perform a nailing operation, the robot needs to reach, nail, and withdraw. Same as the upper-layer array, Cell RT and RR represent the "At" relationship and "Withdraw". Cell TT indicates whether the motion has been completed by the assigned motion ID (e.g., for connection operations, nail is 1 and chalk is 2). The provided standard array supports low-level operations that consist of up to three primitives: approaching certain locations, withdrawing, and one additional skill primitive. To further improve the flexibility of handling special low-level operations that require more than three or other types of primitives (e.g., the floor underlayment connection operation requires the robot to reach the attaching location, flatten the underlayment, attach the underlayment, and withdraw), the bottom-layer array structure can be customized. The customization can be based on the specific needs of the operation, either by defining extra cells or by modifying the functions to compute cell values. In cases when a special bottom-layer array is required by an operation, the robotic engineer will program the reasoning primitive to initiate the operation as generating the customized bottom-layer array instead of the standard one. As a result, the operation can be integrated into the construction workflow for automatic sequencing. For SDAs on different layers, the probabilistic mappings are saved separately in the robot knowledge base.

For the three high-level operations, connection (CC), material preparation (MM), and target finishing (TT), multiple layers of arrays are involved to determine the elemental motion sequences. Transitions between different layers are facilitated by reasoning primitives following the process in Fig. 4. The reasoning primitives can retrieve BIM data (e.g., number of operations) to generate SDA or switch to another layer. During human demonstrations, reasoning primitives can be selected from the GUI of the digital twin system.

The system incorporates six default transitions that are activated when specific conditions are met. First, if values in the transit-layer or bottom-layer SDA are all zero, signifying the completeness of the operations, the system will switch to the upper layer. Second, if there is only a solitary row present in the transit-layer SDA, revealing the requirement of just one type of low-level operation, the system will by default directly begin the operation. Third, regardless of the number of identical low-level operations (e.g., 4 screws) needed in a transit-layer SDA, only the array of a single operation (e.g., 1 screw) is created at a time. Once the operation is completed, the system reverts to the transit layer to verify if all operations of the same type have been executed. If not, the system will, by default, automatically proceed with this type of

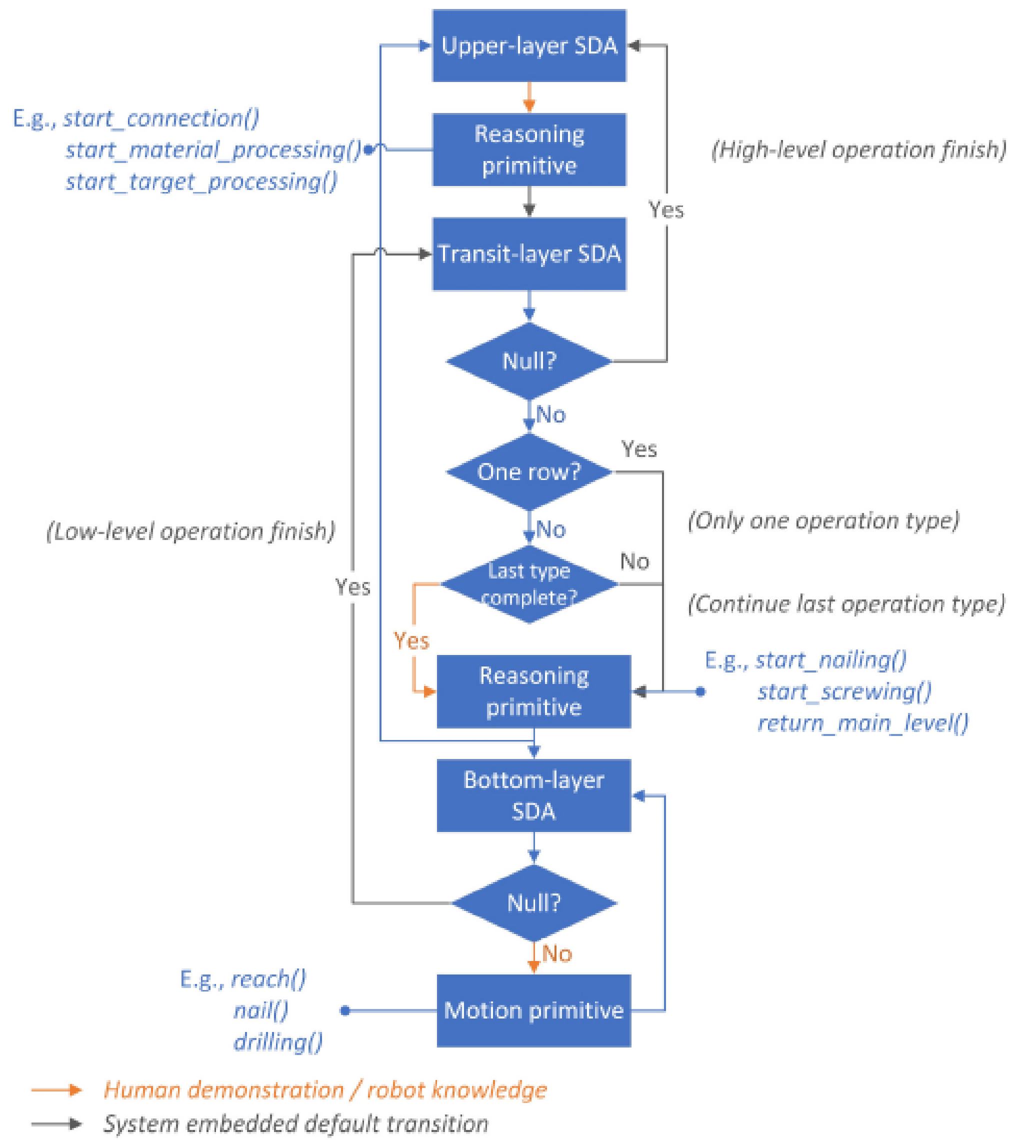

Fig. 4. Decision process for motion sequences of high-level operations.

operation until such operations are finished. This method eliminates the need for repeated demonstrations when multiple same type operations are required for workpiece assembly. Sequencing skills learned can be leveraged to automatically establish the motion sequence for subsequent operations of the same type. Finally, the start transition retrieves the next target from the BIM and generates the initial upper-layer SDA, and the finish transition concludes the SDA to advance to the next target.

The digital twin system incorporates the arrays and functions that support their generation and inter-layer transition. It also provides an intuitive user interface, making it accessible to construction workers with minimal training. When developing motion primitives, three factors must be taken into account: (1) the robot or machine designated for task execution; (2) the required BIM and sensor data as input for planning and execution, along with the process of requesting the information; (3) the method to control the robot given the tools and information required. By intuitively selecting encapsulated skill primitives through the LfD module with the digital twin system, human workers can impart motion sequencing skills for different construction tasks to robots. After learning, the robot can autonomously determine the motion sequence under human supervision, eliminating the need to program for each task variation or step-by-step human guidance.

4. Experimental case studies

Case studies featuring three representative scenarios in construction assembly (exterior wall sheathing, drywall installation, and timber frame construction), are presented to verify the proposed approach. The objective of this section is to present in detail how SDAs change with the construction scene and decide the motion sequence and to demonstrate how learned knowledge is reused for the same task and is transferred to a different construction task. Therefore, we selected exterior wall sheathing and drywall installation as the first two scenarios, since they have many similarities, and a third scenario of timber frame construction that is more complex than the other two. The number of human demonstrations requested and the number of automatic decisions made by the robot throughout the three assembly processes are investigated. Following the incorporation of the LfD module, the digital twin system retains the workflow and functionalities outlined in [20]. Considering

Table 8
List of Skill Primitives.

Motion Primitives	Reasoning Primitives
Reach(P)	start_connection()
Grasp()	start_material_processing()
Release()	start_nailing()
Withdraw()	start_screwing()
Nail()	start_cutting()
Screw()	start_drilling()
Cut(C)	return_upper_layer()
Drill(D)	

that this study primarily focuses on motion sequencing, only processes and information in the digital twin system that are pertinent to motion sequencing and LfD are discussed.

Two robotic arms and a CNC machine form the robotic system in the case studies. One robotic arm manages workpiece manipulation, while the other assists in processing and connection tasks like drilling and screwing, with different tools attached to its end-effector. The tool can either be manually switched by a human co-worker or automatically switched by an automatic tool changer [75] or a quick changer [76]. Workpieces can be cut into different shapes by the CNC machine according to predefined cutting planes. Table 8 lists the skill primitives provided to the robotic system. The robotic arms can generate collision-free motion plans to reach different poses with MoveIt [77,78]. The withdrawal of the robotic arm is predefined. The knowledge base is divided into several sessions, denoted as *M* (upper-layer), *TM*(transit-layer material preparation), *TC*(transit-layer connection), *C*(bottom-layer connection operations), and *MP*(bottom-layer material preparation operations). The information needed for the task and detailed motion planning, such as the construction sequence, targeting poses, and cutting planes, is retrieved from a BIM, which can be generated by computational design or manually defined [61]. In this case study, only material preparation and connection are used as examples of high-level operations. Nevertheless, the system is capable of handling target finishing.

4.1. Exterior wall sheathing

In the first scenario, the robot has just been deployed on the construction site to perform the exterior wall sheathing task. No prior knowledge about construction assembly exists in the robot knowledge base. Data related to the next sheathing target, including material gripping pose sequence $M(m_0, m_1)$, target placing pose sequence $P(p_0, p_1, p_2)$ and nailing pose indicators $N(n_0, n_1, n_2, n_3)$ can be accessed from a BIM (Fig. 5). Noted that instead of using a single 6DOF pose, material gripping and target placing contain a series of poses following manipulation convention. This is to guarantee that the sheathing stays in firm contact with its neighbor. The objective of the task is to install the sheathing at the designated location and fasten it to the frame using nails, as indicated by the GSA in Fig. 6.

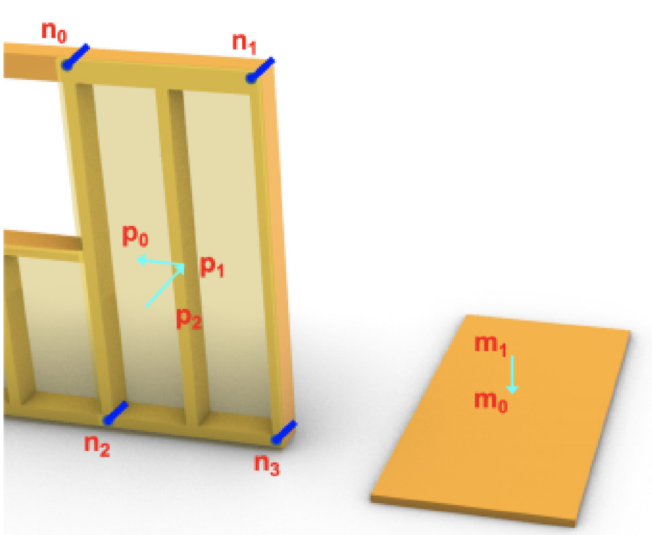

Fig. 5. Illustration of BIM information for Robot Planning.

GSA-1

	M	T	R	C
M	0	1	0	0
T	0	0	0	0
R	0	0	1	0
C	0	0	0	1

Fig. 6. GSA of Scenario 1.

When the task begins, the CSA is initiated (Fig. 7). The SDA is calculated as the distance from CSA to GSA (SDA100). Since SDA100 does not exist in the robot knowledge base, the human co-worker provides demonstrations by selecting *Reach*() from the dropdown menu in the digital twin. Then, they will follow system guidance to select the construction material as the input of *Reach*(). The pose sequence *M* to grip the selected material is retrieved from the BIM. After the motion plan is developed and approved by the human co-worker, *Reach*(*M*) is executed. Mapping from SDA100 to *Reach*(*M*) is recorded as upper-level mapping knowledge (U1). Since the robot end-effector is located at the material location after completion of motion *Reach*(*M*), value of Cell RM changes to 1 and SDA is updated accordingly (SDA101). Next, *Grasp*() primitive is selected by the human co-worker to request the robot to grab the material workpiece. As a result, the material is on the robot, and Cell MR updates to 1. Mapping from SDA101 to *Grasp*() is saved as U2 in the knowledge base.

After the material is picked up, the robot is guided by the human co-worker to bring the material to the target place. After *Reach*(*P*) is selected, the robot generates a collision-free motion plan to reach the target placing poses $P(p_0, p_1, p_2)$. After the operation, both the robot and the material are located at their corresponding target place, updating Cell MT and RT to 1 (Fig. 8). Then, the connection operation is activated with the *start_connection*() reasoning primitive. The reasoning primitive obtains details of the connection operation from the BIM and creates the transit-layer SDA (SDA104), which indicates 4 nailing operations (ID = 1) are needed. Similar to the motion primitive, mapping from SDA103 to this reasoning primitive is also saved in the upper-level knowledge base (U4).

Since only the nailing operation is needed for connection, the *start_nailing*() reasoning primitive is automatically chosen without requiring demonstration, as specified by the system default transitions. The bottom-layer CSA and SDA for the first nailing operation are created accordingly (Fig. 9). Then, the robot follows a series of human guidance, including reaching the nailing indicator, activating the nail gun, and withdrawing the end-effector to perform the nailing operation. Meanwhile, the bottom-layer SDAs change at each step (SDA106, SDA107, SDA108). The demonstrations are saved as connection mapping (C1, C2, C3) in the knowledge base. After withdrawal, all the values in the bottom-layer SDA become zero. Therefore, the system returns to the

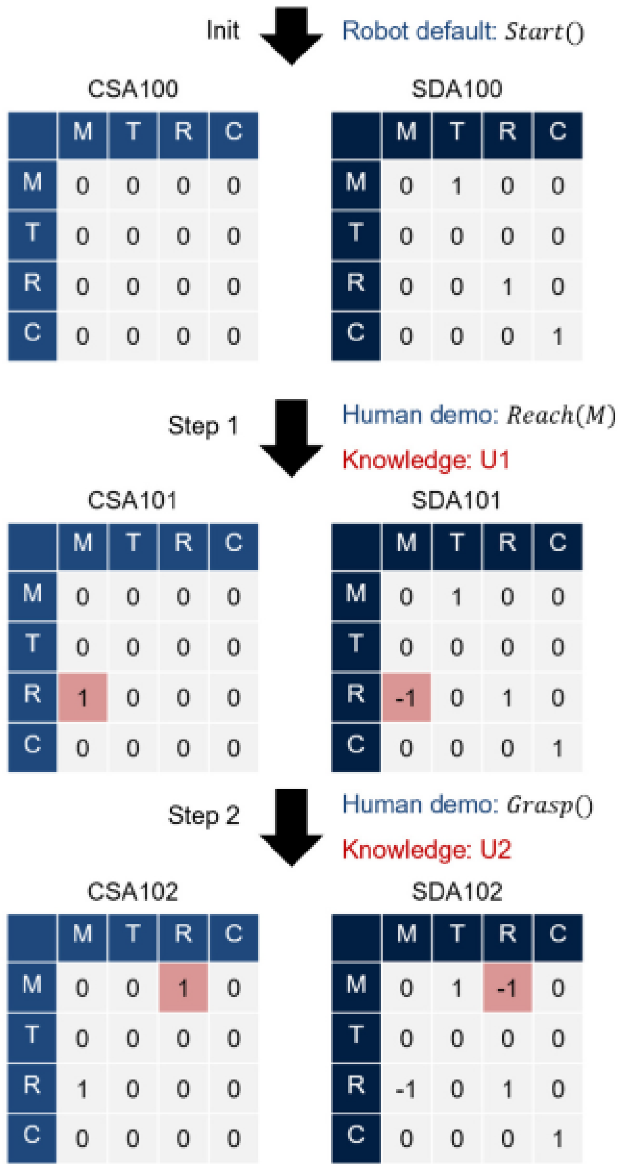

Fig. 7. Scenario 1 Steps 1 and 2 Updates.

transit layer, as defined by default transitions. It then restarts the loop and drives in another nail using learned mappings (C1, C2, C3) until all the required nails are finished.

When all nails are installed, values in the transit-layer SDA are all zeros, signifying the completeness of all connection operations (Fig. 10). Therefore, it returns to the upper layer. Human co-workers continue to guide the robot to release and withdraw its gripper (Fig. 11). After withdrawal, all values in the upper-layer SDA are zero, indicating the current workpiece is fully assembled and installation of the next workpiece in the row can start.

4.2. Drywall installation

After finishing exterior wall sheathing, the robot is assigned to work on drywall installation. Drywall installation bears a resemblance to exterior wall sheathing, with the distinction being that drywall panels are fastened to the frame with screws rather than nails. As the robot has already acquired expertise in exterior wall sheathing, the learned knowledge is applied to automatically decide skill primitives for drywall

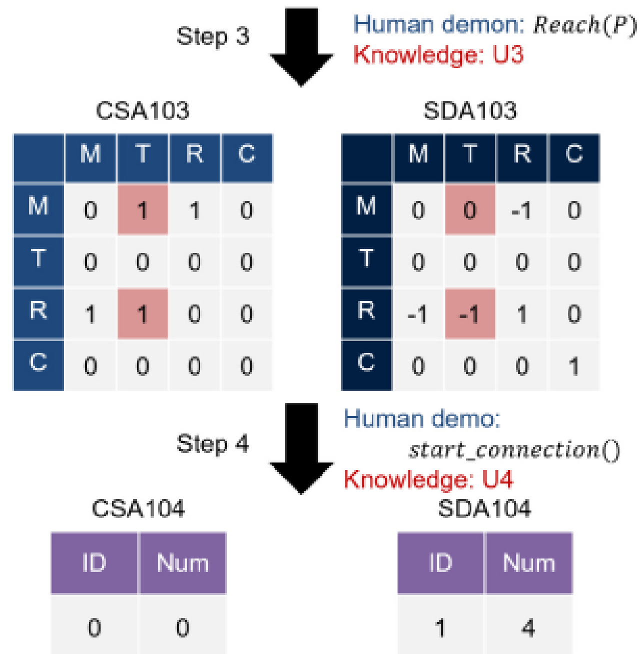

Fig. 8. Scenario 1 Steps 3 and 4 Update.

installation. Similar to Scenario 1, the material gripping pose sequence $M(m_0, m_1)$, target placing pose sequence $P(p_0, p_1, p_2)$, and screwing pose indicators $S(s_0, s_1, s_2, s_3)$ for this scenario are also saved in a BIM.

The initial array of drywall installation is identical to that of exterior wall sheathing. With the learned mapping from the knowledge base, the robot makes decisions for the first four steps, which are consistent with wall sheathing (Fig. 12). However, a new transit-layer with a different ID is created after the connection operation starts because a new type of low-level operation (screwing) is needed for this scenario. Since screwing is the only type of connection operation, it automatically starts screwing and creates the bottom-layer SDA.

Since SDA205 has not been encountered by the robot, the human co-worker needs to provide two demonstrations as new connection mappings, after which the robot can utilize the learned knowledge to automatically decide all the subsequent primitives in the connection operation (Fig. 13). After returning to the upper layer, U5 and U6 mapping in the knowledge base is used to automatically decide the release and withdraw motions to finish the assembly task.

4.3. Timber frame construction

In contrast to the two highly similar scenarios in Section 4.1 and 4.2, assembly of the timber frame has greater complexity. Fig. 14 illustrates the process for this scenario. After the workpiece is picked up following poses $M(m_0, m_1)$ (Fig. 14a), it is brought to the CNC machine for cutting based on the predefined cutting plane C. Then, the workpiece is held in place when being drilled by another robot at $D(d_0, d_1, d_2, d_3)$ (Fig. 14b). Finally, the workplace is placed onto the target position after being manipulated through a sequence of defined poses $P(p_0, p_1, p_2)$ (Fig. 14c) and is screwed onto the wall frame through the drilled holes $S(s_0, s_1)$ (Fig. 14d).

Different from the two previous scenarios, Cell MM in the GSA is 1 because some material preparation operations are needed (Fig. 15). As a result, the initial SDA has not been learned by the robot, and human demonstrations are required to guide the robot to reach and grasp the stud material. At Step 3, the material preparation operation is initiated by the human co-worker (Fig. 16). In this case, two low-level operations, cutting (ID = 1) and drilling (ID = 2) are needed, each representing a

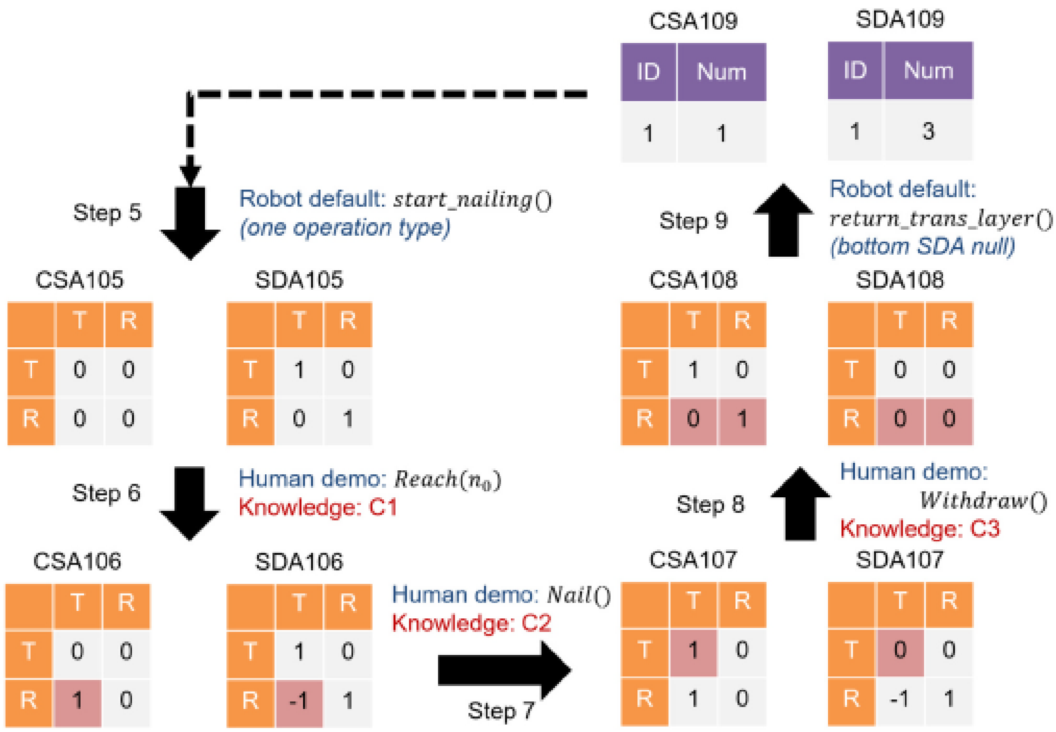

Fig. 9. Scenario 1 Nailing Operation Updates.

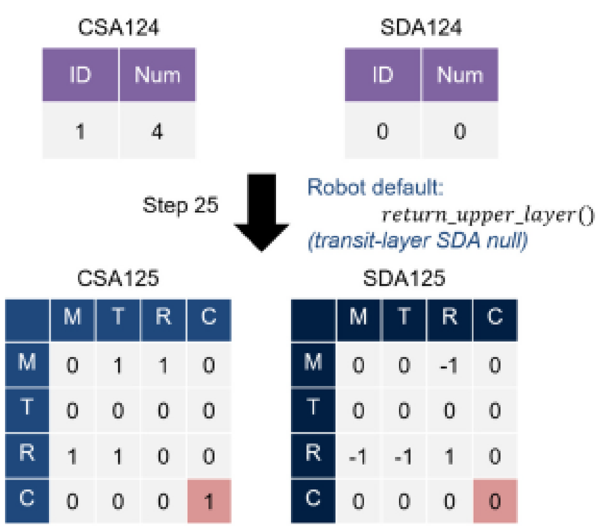

Fig. 10. Scenario 1 Step 25 Update.

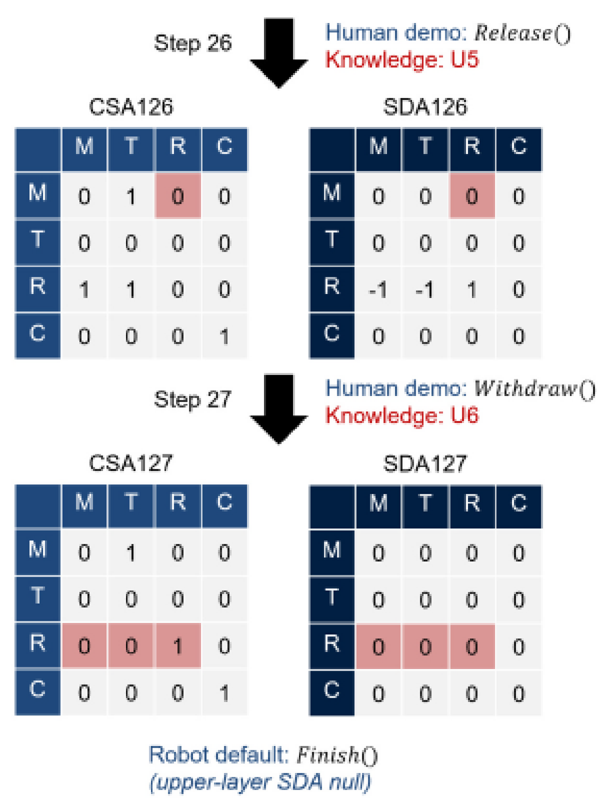

Fig. 11. Scenario 1 Step 26 and 27 Updates.

row in the transit-layer SDA. Therefore, the low-level operation to start first needs to be selected by the human co-worker as a demonstration. Once the cutting operation is finished, values on the corresponding row in the transit-layer SDA change to zero (Fig. 17). As a result, the robot automatically initiates the drilling operation because there is only one type of low-level operation remaining in the transit-level array. Other steps are similar to the previous scenarios and thus are not listed.

5. Proof-of-concept implementation

The scope of this study is the interaction, learning from demonstration, and skill and scene representation techniques to enable the automatic motion sequencing process. To verify that this objective can be achieved with the proposed system, we explored a wooden frame construction activity in simulation as a proof-of-concept implementation. The experimental platform is shown in Fig. 18. The construction task is performed by a Kuka KR120 industrial robotic arm. Its end effector can

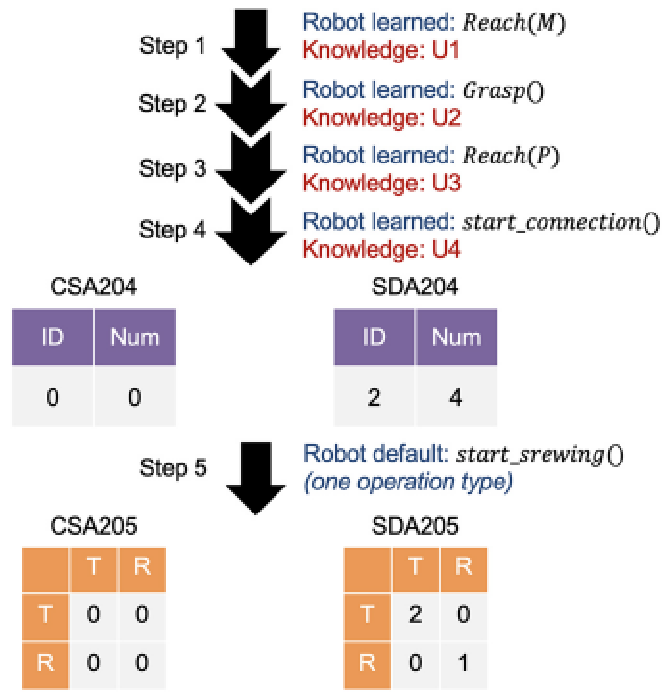

Fig. 12. Scenario 2 Steps 1 to 5 Updates.

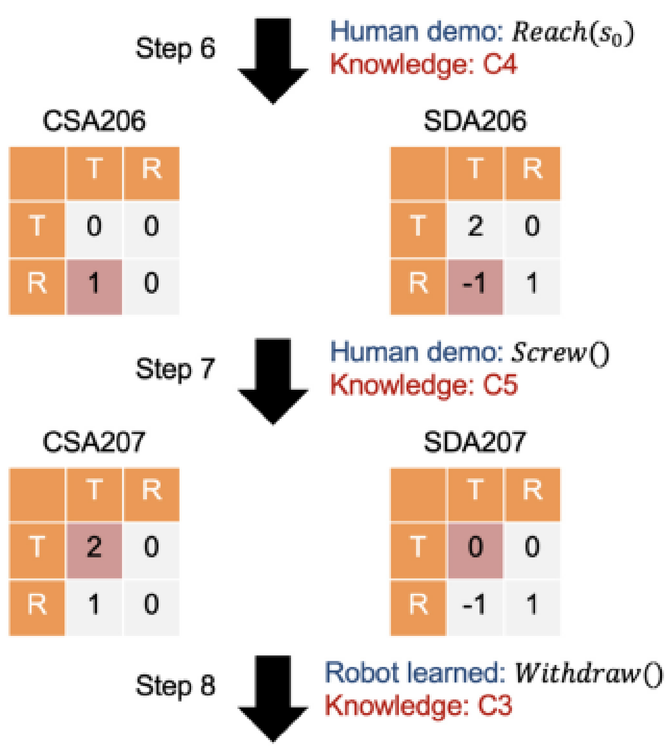

Fig. 13. Scenario 2 Steps 5 to 7 Updates.

both grip and nail by combining a vacuum gripper and a nail gun. The digital twin for HRCC is developed with Unity (user interface) and ROS (middleware). Construction-related information is saved in a BIM in Rhino (Fig. 19), which can communicate with both Unity through Grasshopper using Rhino.Inside and with ROS using Rosbridge [79]. There are two virtual robot models in Unity that can be controlled with joint state data from ROS. The virtual "execution" robot is synchronized with the robot in Gazebo to reflect robot status in ROE, while the virtual

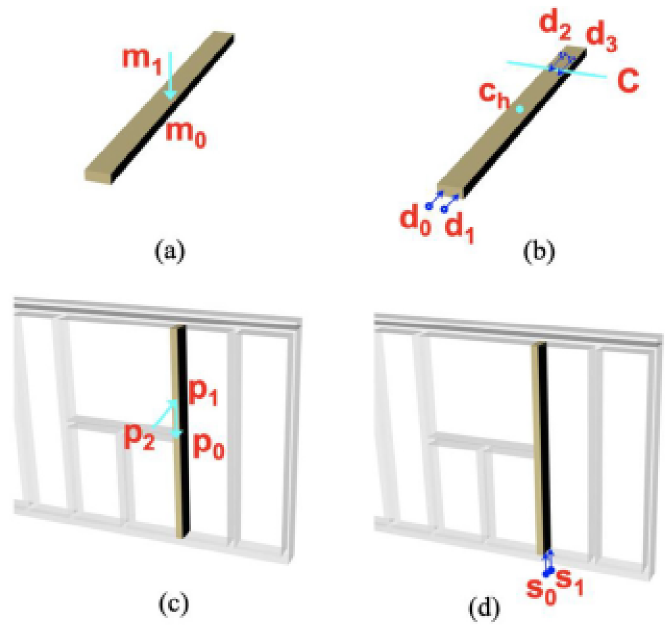

Fig. 14. Timber frame construction workflow.

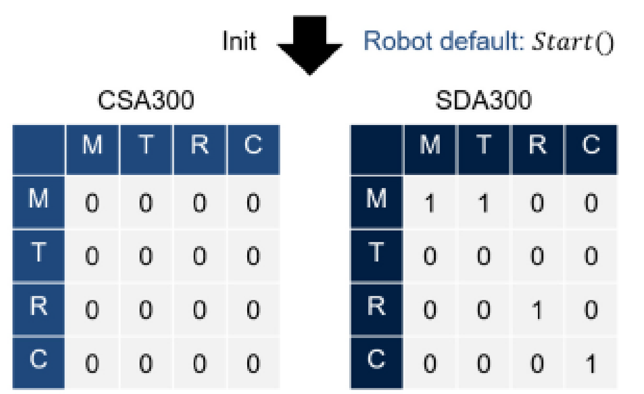

Fig. 15. Scenario 3 Initial States.

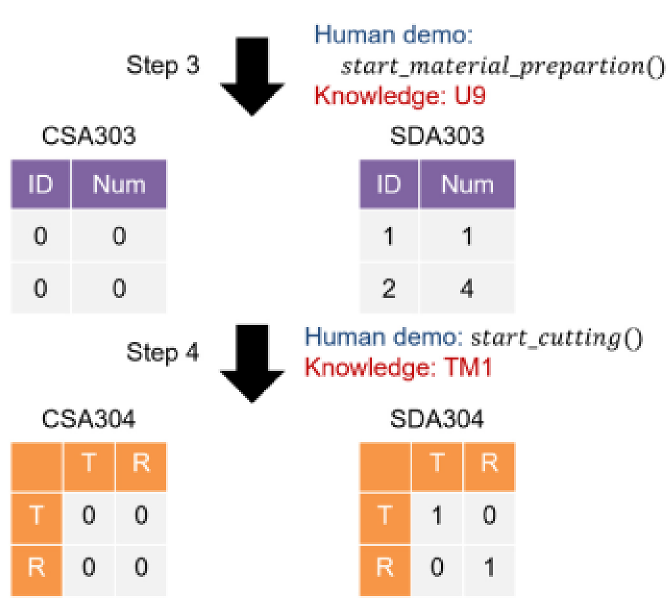

Fig. 16. Scenario 3 Steps 3 and 4 Updates.

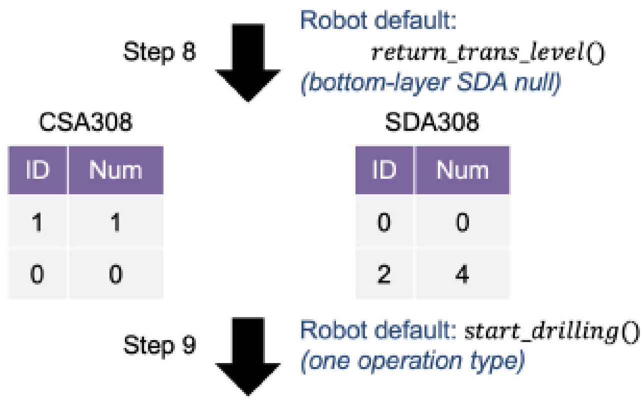

Fig. 17. Scenario 3 Steps 8 and 9 Updates.

"planning" robot is used to demonstrate plan preview to human co-workers as animations and appears only when needed. The robot model in ROS is used for motion planning.

The ROE is simulated in Gazebo. Simulation enables the testing and evaluation of emerging robotic systems in a manner that is both safe and economical, significantly mitigating the associated risks and costs. It has been adopted by several studies to verify the feasibility of new proposed robotic methods (e.g., [13,80,81]). Gazebo is a robust physical engine that allows rapid prototyping of robotic tasks and is considered a validated digital twin to evaluate new robotic methods such as those proposed in this study within a virtual environment [82,83]. Through ROS and Automation Device Specification (ADS), movements of the robot emulator in Gazebo can subsequently be synchronized with the physical robot in real time [20,60], which enables Gazebo-based simulations to be widely employed as the prototype to assess the feasibility of newly proposed methods [84,85]. Although there are uncertainties and challenges such as arm control synchronization or localization errors for a real robot to perform construction work, these potential differences between simulation and real robot execution do not diminish the contribution or affect the validity of the proposed method for motion sequencing. The authors have other studies specifically addressing robot synchronized control between Gazebo and a real robot [60], physical robotic system setup [20], and adaptive control [86] for real robot execution.

The robot will first install the two studs at the bottom, followed by

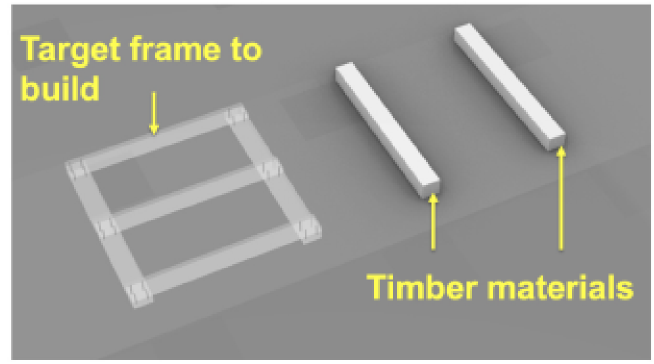

Fig. 19. BIM of the wooden frame construction acitivity.

Table 9
BIM Component Data Structure.

Name	Name of the component
Family	The category of the component. Options are Workpiece, Connection (e.g., position indicator of a screw), or Processing (indicator of a cutting plane).
Parents	The component the target is associated with. Optional for Workpiece and mandatory for Connection or Processing families.
Type	Corresponding material type.
Pose	A list of 6DOF poses containing a sequence of poses to reach the component.
Order	Construction sequence
Connection	Apply to components in the Workpiece family. It is a list of connection operations associated with the component (e.g., [nailing, screwing]).
Processing_M	Apply to components in the Workpiece family. It is a list of material preparation operations associated with the component (e.g., [drilling, cutting]).
Processing_T	Apply to components in the Workpiece family. It is a list of target finishing operations associated with the component (e.g., [painting, polishing]).
Methods	Mandatory for components in the Connection or Processing family. It indicates the corresponding operation method of the component. For example, for a screwing point indicator, the method should be "screwing".

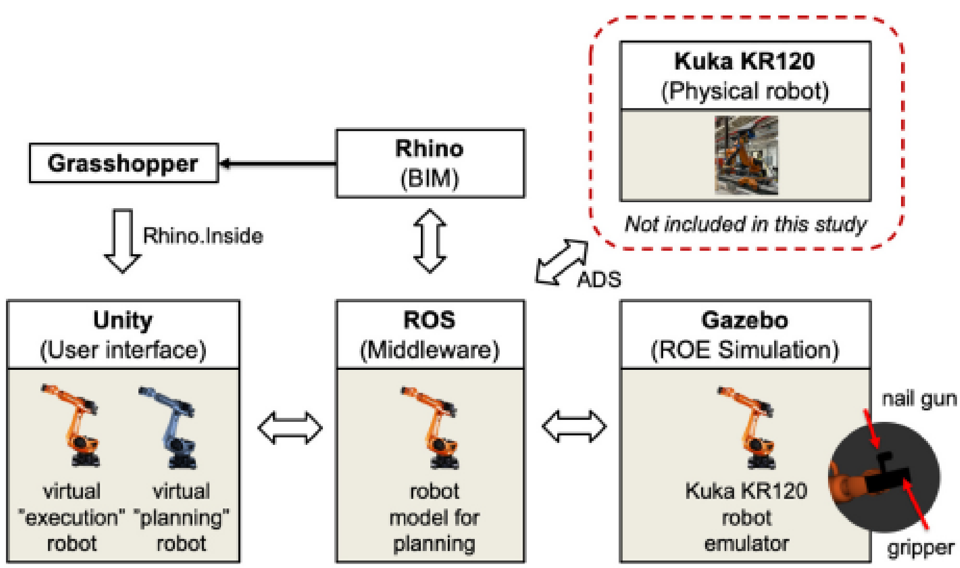

Fig. 18. The experimental platform for wooden frame installation experiment.

the three studs on the top. Eight nails (four on each side) are used to secure each top stud to the bases. A data structure to store attribute information in BIM has been proposed (Table 9). Compared to the data structure in [20], the new version supports more intricate construction tasks involving diverse operations (e.g., connection).

Calculation for the "At" relation requires the 3D Euclidean distance to be shorter than 0.01 m for two objects to be considered at the same position. When determining whether the robot end-effector reaches an object, the robot end-effector position is offset based on the gripper shape to calculate the Euclidean distance to the object. At the same time, all the roll, pitch, and yaw orientation differences need to be smaller than 0.005 rad. The expression to determine the "At" relationship with the threshold coefficient is shown in Eq. 3.

$$At(A,B) = \begin{cases} 1, & \sqrt{(x_A-x_B)^2+(y_A-y_B)^2+(z_A-z_B)^2} \leq 0.01\text{m} \\ & \wedge|\varphi_A-\varphi_B| \leq 0.005 \text{ rad} \\ & \wedge|\theta_A-\theta_B| \leq 0.005 \text{ rad} \\ & \wedge|\psi_A-\psi_B| \leq 0.005 \text{ rad} \\ 0, & \text{else} \end{cases} \tag{3}$$

for two objects $A(x_A, y_A, z_A, \varphi_A, \theta_A, \psi_A)$ and $B(x_B, y_B, z_B, \varphi_B, \theta_B, \psi_B)$, where x, y, z denote positions and φ, θ, ψ denote the roll, pitch, and yaw of the object.

Throughout the HRCC process, the human co-worker uses the digital twin interface to monitor the ROE and interact with the robot. After the target component and its pose suggested by the BIM are confirmed by the human co-worker (Fig. 20a), the SDA is created and the LfD process begins. For unknown SDAs, human demonstrations will be requested by the robot. The digital twin provides the interface for humans to choose corresponding primitives (Fig. 20b). Furthermore, the human co-workers will be prompted to demonstrate the input related to the selected primitive, if applicable (Fig. 20c). Otherwise, the robot displays its decisions in the digital twin for human approval if the SDA exists in the robot's knowledge base (Fig. 20d).

In cases where the chosen primitive is a motion primitive that requires motion planning, such as *Reach*, a detailed motion plan generated by the robot will be previewed by the human co-worker. They can either approve the plan for execution or request the robot to devise an alternative if they are unsatisfied with the demonstrated one. When a reasoning primitive is selected, the SDA is automatically updated. Furthermore, if human co-workers are unhappy with the primitive selection, they can opt for a different primitive. A video demonstrating the LfD process is uploaded as Supplementary Data.

6. Results and discussion

For a robot to successfully perform construction activities, a variety of technological advancements (e.g., perception, localization, hardware design, control) need to come together. Instead of attempting to address all these challenges at once, this study focuses on a specific aspect of how to enable automatic motion sequencing for multi-task construction robots. Therefore, three case studies and a wooden frame construction activity are designed. Eight motion primitives, seven reasoning primitives, and six default transitions are provided. The objectives of these experiments are: (1) to evaluate and characterize how this workflow reduces human effort in guiding the robot to perform a construction activity; and (2) to characterize how this workflow allows the knowledge learned to be reused by the robots across multiple construction activities.

In order to understand the robot learning performance, the efficiency of human demonstration, and the quality of the defined default transitions, four evaluation metrics are proposed and used, including (1) teaching effort: the percentage of demonstrations needed from the human co-worker out of steps other than default transitions (Eq. 4); (2) teaching quality: the percentage of decisions made by the robot using knowledge learned from humans out of steps other than default transitions (Eq. 5); (3) teaching efficiency: the number of robot decisions from learned knowledge to the number of human demonstrations ratio (Eq. 6); and (4) default transition quality: the percentage of decisions made by the robot using the six defined default transitions introduced in

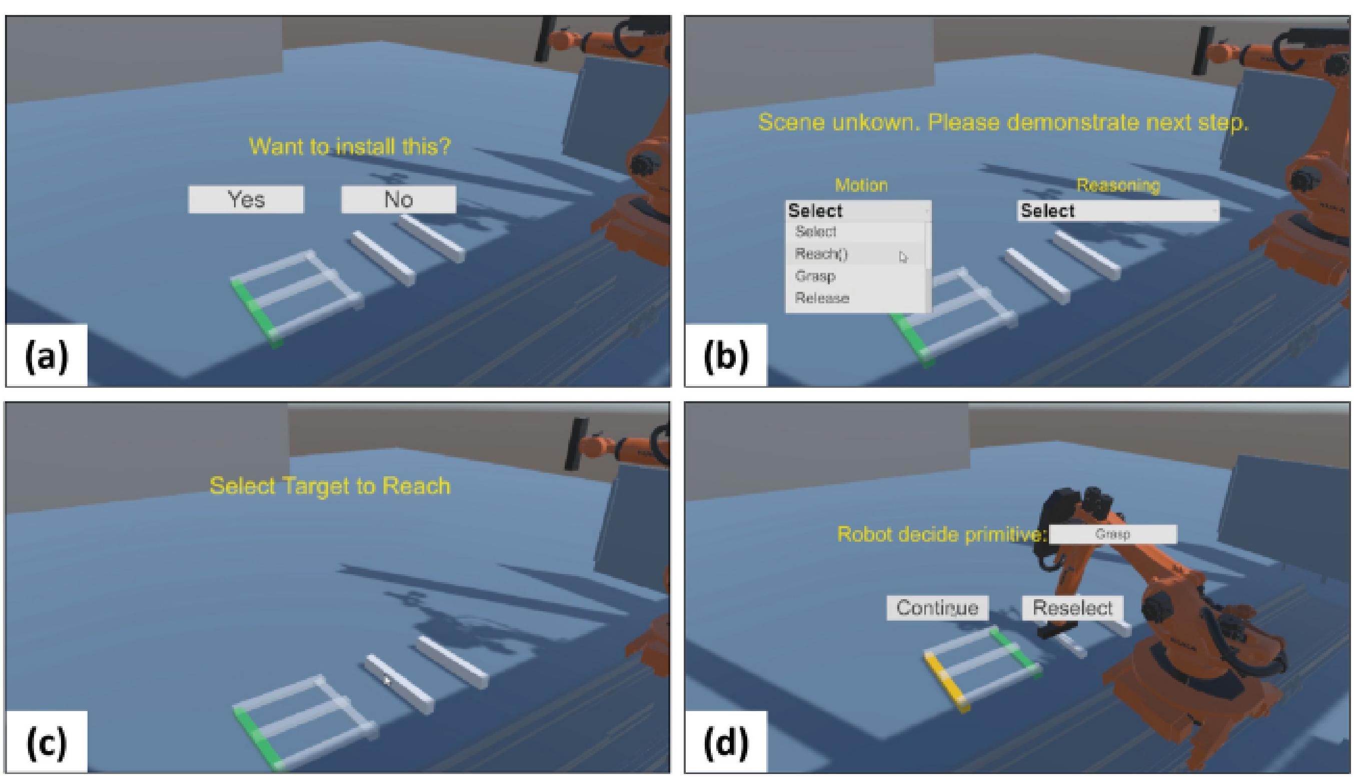

Fig. 20. Digital twin LfD interface.

Section 3.4 out of all steps (Eq. 7) [38]. Num_{Demo}, $Num_{Learned}$, and $Num_{Default}$ in the equations represent the number of steps decided by human demonstrations, robot learned knowledge, and default transitions respectively. The knowledge transfer is evaluated by analyzing the trend of these metrics across tasks.

$$\text{Teaching effort} = \frac{Num_{Demo}}{Num_{Demo} + Num_{Learned}} \times 100\% \tag{4}$$

$$\text{Teaching quality} = \frac{Num_{Learned}}{Num_{Demo} + Num_{Learned}} \times 100\% \tag{5}$$

$$\text{Teaching efficiency} = \frac{Num_{Learned}}{Num_{Demo}} \tag{6}$$

$$\text{Default transitions quality} = \frac{Num_{Default}}{Num_{Demo} + Num_{Learned} + Num_{Default}} \times 100\% \tag{7}$$

The human-robot collaborative wooden frame construction process is demonstrated through the proposed digital twin system. Since the robot starts with no prior knowledge, higher teaching effort is needed from the human co-workers at the onset. The teaching effort is 100% (5 steps) for the first stud on the base (Fig. 21). For the first stud on the top, the teaching effort reduces to 27.59% (8 steps) even though the procedures are different, which indicates that the robot has effectively learned from a different previous task. After learning, the robot can automatically decide the whole sequence of primitives to install the second or third stud (i.e., 0% teaching effort). More specifically, for the top studs, after the motion sequence to perform the first nailing operation on the first stud is learned, the robot can take advantage of the learned knowledge and automatically perform all the remaining nailing operations.

Installation for the studs on the top is more complicated because connection operations with nails are needed. The provided default transitions play an important role in enabling the high autonomy level for complex operations such as this. There is a decrease in teaching efficiency when the system performs the new task of top stud installation for the first time. However, when it repeats the previous tasks, the teaching efficiency is extremely high since no human demonstration is needed. In total, the robot makes 91.77% (53.16% from learned knowledge and 39.87% from default transitions) of decisions during the overall frame construction process, and the human worker only teaches 8.23% of the total steps. These experiments demonstrate that the system not only allows humans and robots to collaboratively sequence the given primitives to perform construction work, but also reduces human effort and increases system autonomy by enabling robot learning.

The results for the three case study scenarios are shown in Table 10. Despite the limited number of default transitions provided, they constitute a significant portion (37.93% or 37.50%) of the decisions to assist in the transitioning between different layers. For the first scenario, human teaching effort is 50% (9 steps) to sheath the first panel, and another 50% (9 steps) are solved by robot learned knowledge. As the learned knowledge is taken to the second scenario to perform the drywall installation, the teaching effort is only 11.11% (2 steps) and 16 (88.89%) steps can be solved by robot learned knowledge. The substantial decrease in the proportion of human demonstrations is due to the similarities in the assembly sequences between the two scenarios.

When it comes to the more complicated timber frame construction scenario, the teaching effort increases slightly compared to Scenario 2 as it involves additional material preparation steps such as drilling and cutting, making it more intricate and distinct from the other two scenarios (Fig. 22). Nevertheless, the robot can still solve 74.29% of steps with its knowledge, and the human co-worker only needs to demonstrate 9 (25.71%) steps. Even though the teaching efficiency decreases significantly for Scenario 3 because more demonstrations are needed due to higher task complexity, the teaching quality maintains at a relatively high level since the robot experiences more scenarios. These observations demonstrate that the system can increase motion sequencing autonomy and reduce human effort by transferring learned

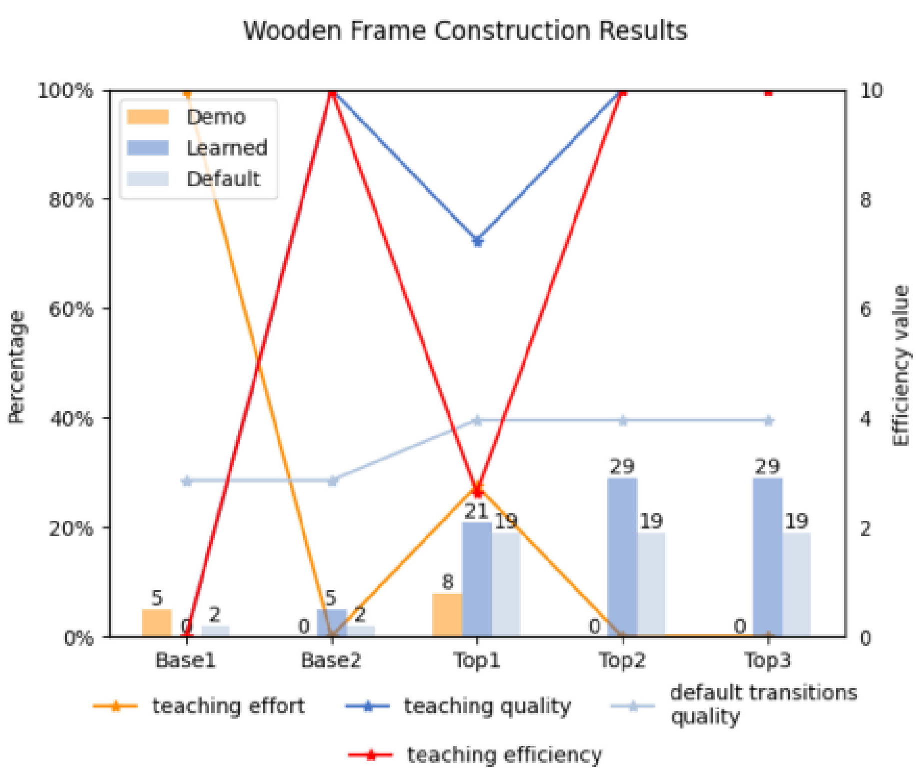

Fig. 21. Human and robot decision distribution for wooden frame construction (Teaching efficiency is presented as 10 when no demonstration is needed).

Table 10
Results of Case Study Scenarios.

	Scenario 1 (Exterior wall sheathing)	Scenario 2 (Drywall installation)	Scenario 3 (Timber frame construction)
Teaching effort (steps)	50.00% (9)	11.11% (2)	25.71% (9)
Teaching quality (steps)	50.00% (9)	88.89% (16)	74.29% (26)
Default transition quality (steps)	37.93% (11)	37.93% (11)	37.50% (21)
Teaching efficiency	1	8	2.89
Total steps	29	29	56

knowledge within the task and to different types of construction activities. It should be noted that we assume there are only four screws or nails for each panel in Scenario 1 and Scenario 2. Real-life construction scenarios typically require a higher number of screws or nails for each panel, resulting in higher autonomy and lower human effort.

Some limitations are observed through the case studies and proof-of-concept implementation. First, even though the robot can automatically make decisions by learning from human demonstrations, we still request the human co-worker to approve each decision before the robot can take any action. This increases human workload and reduces system productivity to some extent when plenty of similar nailing operations are needed and approval is required at each step in the proof-of-concept implementation. This is related to the tradeoff between safety and human workload of supervision. From our observations, human co-workers are more comfortable when they are aware of robot movement at the next step, especially when they are working with the actual robot. Thus, the functions of previewing and approving plans are provided. In the future, human co-workers' supervision workload can be optimized by investigating their trust in the robot decisions under different situations and dynamically encapsulating primitives. Second, it takes effort to prepare the shop drawing model and semantic information in the BIM to drive the construction process [87]. In the future, computational design should be considered to automate the process to prepare the BIM for the system, such as automatically generating nailing positions, calculating cutting planes, and adding attributes to components [61,88]. Third, the proposed method mainly focuses on the sequencing of primitives instead of the execution of a specific operation. The capability of the system for specific construction applications also depends on the capability of the skill primitives known to the robot. However, preprogrammed, autonomous, or learned skills from human demonstration can be conveniently added as skill primitives for the robot to work on broader construction assembly applications in the future.

Compared to other state-of-the-art methods for motion sequencing, the proposed method has several distinct characteristics to support complex construction assembly tasks while reducing human effort and computation resources. First, the current task status is tracked with the digital twin system and represented by multi-layer SDAs, which are better at handling complex construction assembly scenarios with preparation, connection, and finishing operations compared to vision-based state detection [37,39]. The robot plans based on the detected environment states instead of assuming that all prior motions are completed successfully, making it more robust to uncertainties [38,49]. Second, the system requires fewer demonstrations and computation resources compared to methods based on machine learning or the Markov decision process [48,50,52–54]. The system only requires the human co-worker to demonstrate the task once to build the corresponding SDAs and learn the sequencing knowledge. Even within one construction task, the human co-worker does not need to demonstrate the whole process. For example, when one nailing operation is demonstrated, the sequence to

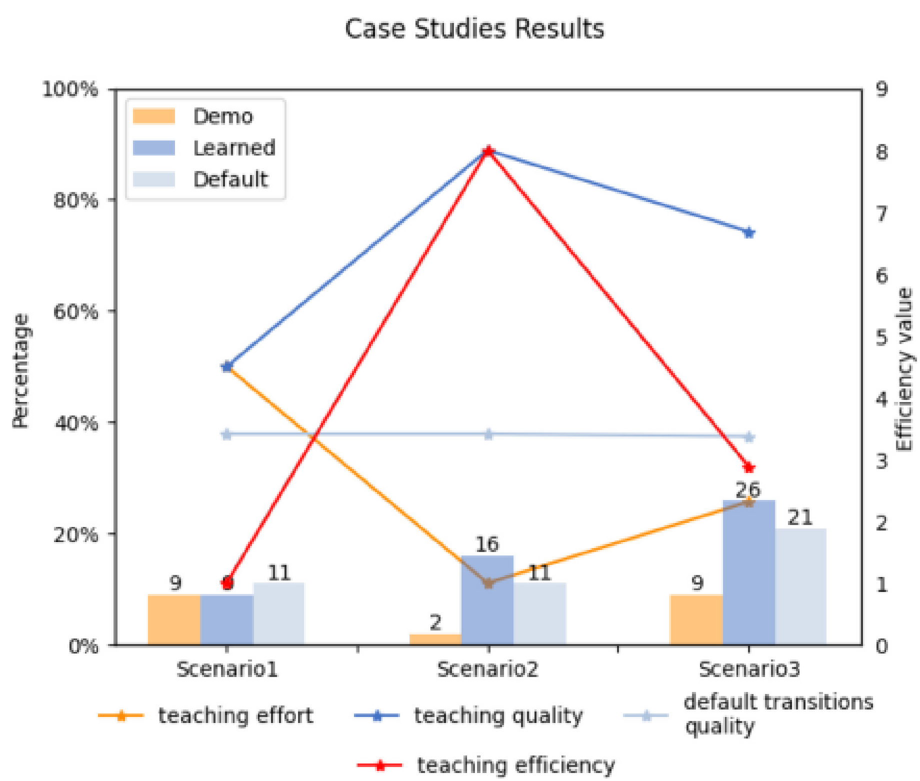

Fig. 22. Human and robot decision distribution for case studies.

install other nails can be automatically generated. Third, the system supports transferring the learned knowledge within and across different construction tasks. By extracting workpieces one by one from the construction sequence in the BIM, the status modeling SDA focuses on one workpiece at a time. As a result, the similarity and repeatability between tasks and activities increase, allowing efficient transfer of the knowledge within the task (e.g., apply the knowledge for one operation to other operations) and across different activities (e.g., use the knowledge from exterior wall sheathing to reduce the teaching effort for drywall installation).

7. Conclusions

This paper aims to investigate robot automatic motion sequencing so that they are capable of multiple construction assembly tasks and can keep expanding their skillset. A process-level digital twin is used for the robot to determine motions from the task objectives and scene state or to request human demonstrations for unencountered situations. Not only is the acquired knowledge applicable to the same types of construction tasks, but it can also be transferred to other tasks to minimize the amount of teaching required.

The research makes several notable contributions. First, it establishes a four-level taxonomy—elemental motions, operations, tasks, activities—and uses skill primitives to facilitate modular construction robot “programming”, making robots accessible for construction workers. Second, multi-layer SDAs are used to effectively save the learned knowledge as probabilistic mapping and retrieve the knowledge for automatic motion sequencing. From three case studies and the wooden frame construction activity, we observed that robots have higher autonomy and are less dependent on human demonstrations after learning a variety of construction tasks. Third, based on the automatic motion sequencing technique developed, we conceive a construction robot delivery framework that bridges the domain knowledge gap between robotics and construction while improving construction robot work quality and cost efficiency. This research has the potential to enable the widespread application of robots in construction. In the future, we will optimize the human supervision workload by exploring primitive encapsulation and investigating robot confidence levels in decision making.

Supplementary data to this article can be found online at https://doi.org/10.1016/j.autcon.2023.105071.

Declaration of Competing Interest

Carol C. Menassa reports financial support was provided by National Science Foundation. Vineet R. Kamat is a member of the Editorial Board for this journal Automation in Construction.

Data availability

Data will be made available on request.

Acknowledgments

The work presented in this paper was supported financially by two United States National Science Foundation (NSF) Awards: 2025805 and 2128623. The support of the NSF is gratefully acknowledged. Any opinions, findings, conclusions, or recommendations expressed in this paper are those of the authors and do not necessarily reflect the views of the NSF.

References

[1] Mckinsey Global Institute, Reinventing construction through a productivity revolution, McKinsey (2017). https://www.mckinsey.com/business-functions/operations/our-insights/reinventing-construction-through-a-productivity-revolution (accessed March 21, 2022).

[2] J.M. Davila Delgado, L. Oyedele, A. Ajayi, L. Akanbi, O. Akinade, M. Bilal, H. Owolabi, Robotics and automated systems in construction: understanding industry-specific challenges for adoption, journal of building, Engineering. 26 (2019), 100868, https://doi.org/10.1016/j.jobe.2019.100868.

[3] K.M. Lundeen, V.R. Kamat, C.C. Menassa, W. McGee, Scene understanding for adaptive manipulation in robotised construction work, Autom. Constr. 82 (2017) 16–30, https://doi.org/10.1016/j.autcon.2017.06.022.

[4] AGC, 2020 Construction Outlook Survey Results National Results. https://www.agc.org/sites/default/files/Files/Communications/2020_Outlook_Survey_National.pdf, 2020.

[5] S. Park, H. Yu, C.C. Menassa, V.R. Kamat, A comprehensive evaluation of factors influencing acceptance of robotic assistants in field construction work, J. Manag. Eng. 39 (2023), https://doi.org/10.1061/JMENEA.MEENG-5227.

[6] X. Wang, X.S. Dong, S.D. Choi, J. Dement, Work-related musculoskeletal disorders among construction workers in the United States from 1992 to 2014, Occup. Environ. Med. 74 (2017) 374–380, https://doi.org/10.1136/OEMED-2016-103943.

[7] X. Xu, B. Garcia de Soto, On-site autonomous construction robots: a review of research areas, technologies, and suggestions for advancement, in: Proceedings of the 37th International Symposium on Automation and Robotics in Construction (ISARC), 2020, https://doi.org/10.22260/isarc2020/0055.

[8] C.-J. Liang, X. Wang, V.R. Kamat, C.C. Menassa, Human–Robot collaboration in construction: classification and research trends, J. Constr. Eng. Manag. 147 (2021) 03121006, https://doi.org/10.1061/(ASCE)CO.1943-7862.0002154.

[9] Dusty Robotics, Build Better with BIM-Driven Layout. https://www.dustyrobotics.com/, 2021 (accessed March 21, 2022).

[10] Construction Robotics, SAM - Bricklaying made simpler and safer. - Construction Robotics. https://www.construction-robotics.com/sam-2/, 2023 (accessed December 13, 2022).

[11] Advanced Construction Robotics, TyBot | The Rebar-Tying Robot. https://www.constructionrobots.com/tybot, 2023 (accessed December 13, 2022).

[12] Hilti, Jaibot Drilling Robot - Hilti USA. https://www.hilti.com/content/hilti/W1/US/en/business/business/trends/jaibot.html, 2023 (accessed December 13, 2022).

[13] S. Kim, M. Peavy, P.C. Huang, K. Kim, Development of BIM-integrated construction robot task planning and simulation system, Autom. Constr. 127 (2021), 103720, https://doi.org/10.1016/J.AUTCON.2021.103720.

[14] B.R. Azar, V.R. Kamat, Earthmoving equipment automation: a review of technical advances and future outlook, J. Informa. Technol. Construct. 22 (2017) 247–265. http://www.itcon.org/2017/13.

[15] A.M. Alashwal, H. Abdul-Rahman, Developing a Conceptual Framework of Fragmentation in Construction, Department of Quantity Surveying, Faculty of Built Environment, University of Malaya, 2014.

[16] L. Huang, Z. Zhu, Z. Zou, To imitate or not to imitate: boosting reinforcement learning-based construction robotic control for long-horizon tasks using virtual demonstrations, Autom. Constr. 146 (2023), 104691, https://doi.org/10.1016/J.AUTCON.2022.104691.

[17] C.-J. Liang, V.R. Kamat, C.C. Menassa, W. Mcgee, Trajectory-based skill learning for overhead construction robots using generalized cylinders with orientation, J. Comput. Civ. Eng. 36 (2021) 04021036, https://doi.org/10.1061/(ASCE)CP.1943-5487.0001004.

[18] X. Wang, C.-J. Liang, C. Menassa, V. Kamat, Real-time process-level digital twin for collaborative human-robot construction work, in: Proceedings of the 37th International Symposium on Automation and Robotics in Construction (ISARC), 2020, https://doi.org/10.22260/isarc2020/0212.

[19] T. Bock, Construction robotics, Auton. Robot. 22 (2007) 201–209, https://doi.org/10.1007/s10514-006-9008-5.

[20] X. Wang, H. Yu, W. McGee, C.C. Menassa, V.R. Kamat, Enabling BIM-Driven Robotic Construction Workflows with Closed-Loop Digital Twins Xi, 2023, pp. 1–34, https://doi.org/10.48550/arXiv.2306.09639.

[21] H. Yu, V.R. Kamat, C.C. Menassa, W. McGee, Y. Guo, H. Lee, Mutual physical state-aware object handover in full-contact collaborative human-robot construction work, Autom. Constr. 150 (2023), 104829, https://doi.org/10.1016/j.autcon.2023.104829.

[22] H. Wu, H. Li, X. Fang, X. Luo, A survey on teaching workplace skills to construction robots, Expert Syst. Appl. 205 (2022), 117658, https://doi.org/10.1016/j.eswa.2022.117658.

[23] A. Kramberger, A. Gams, B. Nemec, D. Chrysostomou, O. Madsen, A. Ude, Generalisation of orientation trajectories and force-torque profiles for robotic assembly, Robot. Auton. Syst. 98 (2017) 333–346, https://doi.org/10.1016/J.ROBOT.2017.09.019.

[24] L. Peternel, T. Petrič, J. Babič, Robotic assembly solution by human-in-the-loop teaching method based on real-time stiffness modulation, Auton. Robot. 42 (2018) 1–17, https://doi.org/10.1007/S10514-017-9635-Z/FIGURES/11.

[25] A. Billard, S. Calinon, Robot Programming by Demonstration, in: *Handbook of Robotics*, Springer, 2008, pp. 1371–1394. https://doi.org/10.1007/978-3-540-30301-5_60.

[26] R. Cubek, W. Ertel, G. Palm, High-level learning from demonstration with conceptual spaces and subspace clustering, in: Proc IEEE Int Conf Robot Autom, 2015, pp. 2592–2597, https://doi.org/10.1109/ICRA.2015.7139548.

[27] N. Figueroa, A.L.P. Ureche, A. Billard, Learning complex sequential tasks from demonstration: A pizza dough rolling case study, in: ACM/IEEE International Conference on Human-Robot Interaction, IEEE, 2016, pp. 611–612, https://doi.org/10.1109/HRI.2016.7451881.

[28] D. Kulić, C. Ott, D. Lee, J. Ishikawa, Y. Nakamura, Incremental learning of full body motion primitives and their sequencing through human motion observation, Int. J. Robot. Res. 31 (2012) 330–345, https://doi.org/10.1177/0278364911426178.

[29] R. Lioutikov, G. Neumann, G. Maeda, J. Peters, Probabilistic segmentation applied to an assembly task, in: IEEE-RAS International Conference on Humanoid Robots, IEEE, 2015, pp. 533–540, https://doi.org/10.1109/HUMANOIDS.2015.7363584.
[30] S. Niekum, S. Osentoski, G. Konidaris, A.G. Barto, Learning and generalization of complex tasks from unstructured demonstrations, in: IEEE International Conference on Intelligent Robots and Systems, IEEE, 2012, pp. 5239–5246, https://doi.org/10.1109/IROS.2012.6386006.
[31] N. Zhang, T. Qi, Y. Zhao, Real-time learning and recognition of assembly activities based on virtual reality demonstration, Sensors. 21 (2021) 1–15, https://doi.org/10.3390/s21186201.
[32] S. Dong, C. Feng, V.R. Kamat, Sensitivity analysis of augmented reality-assisted building damage reconnaissance using virtual prototyping, Autom. Constr. 33 (2013) 24–36, https://doi.org/10.1016/j.autcon.2012.09.005.
[33] M. Pardowitz, S. Knoop, R. Dillmann, R.D. Zollner, Incremental learning of tasks from user demonstrations, past experiences, and vocal comments, IEEE Trans. Syst. Man Cybernet. B Cybernetics. 37 (2007) 322–332, https://doi.org/10.1109/TSMCB.2006.886951.
[34] A.H. Behzadan, V.R. Kamat, Integrated information modeling and visual simulation of engineering operations using dynamic augmented reality scene graphs, J. Informa. Technol. Construct. 16 (2011) 259–278. http://www.itcon.org/2011/17.
[35] V.R. Kamat, J.C. Martinez, Automated generation of dynamic, operations level virtual construction scenarios, J. Informa. Technol. Construct. 8 (2003) 65–84. http://www.itcon.org/2003/6.
[36] H. Ravichandar, A.S. Polydoros, S. Chernova, A. Billard, Recent advances in robot learning from demonstration, Annu Rev Control Robot Auton Syst. 3 (2020) 297–330, https://doi.org/10.1146/annurev-control-100819-063206.
[37] K. French, S. Wu, T. Pan, Z. Zhou, O.C. Jenkins, Learning behavior trees from demonstration, in: Proc IEEE Int Conf Robot Autom, 2019, pp. 7791–7797, https://doi.org/10.1109/ICRA.2019.8794104.
[38] A. Mohseni-Kabir, C. Rich, S. Chernova, C.L. Sidner, D. Miller, Interactive hierarchical task learning from a single demonstration, in: ACM/IEEE International Conference on Human-Robot Interaction, 2015, pp. 205–212, https://doi.org/10.1145/2696454.2696474.
[39] L. She, S. Yang, Y. Cheng, Y. Jia, J.Y. Chai, N. Xi, Back to the blocks world: Learning new actions through situated human-robot dialogue, in: SIGDIAL 2014 - 15th Annual Meeting of the Special Interest Group on Discourse and Dialogue, Proceedings of the Conference, 2014, pp. 89–97, https://doi.org/10.3115/v1/w14-4313.
[40] M. Scheutz, E. Krause, B. Oosterveld, T. Frasca, R. Platt, Spoken instruction-based one-shot object and action learning in a cognitive robotic architecture, Proceedings of the International Joint Conference on Autonomous Agents and Multiagent Systems, AAMAS 3 (1) (2017) 1378–1386. https://dl.acm.org/doi/abs/10.5555/3091125.3091315.
[41] A. Zhu, P. Pauwels, B. De Vries, Smart component-oriented method of construction robot coordination for prefabricated housing, Autom. Constr. 129 (2021), 103778, https://doi.org/10.1016/J.AUTCON.2021.103778.
[42] M. Dawod, S. Hanna, BIM-assisted object recognition for the on-site autonomous robotic assembly of discrete structures, Construct. Robotics 3 (2019) 69–81, https://doi.org/10.1007/S41693-019-00021-9.
[43] 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.
[44] X. Wang, C.-J. Liang, C.C. Menassa, V.R. Kamat, Interactive and immersive process-level digital twin for collaborative human–robot construction work, J. Comput. Civ. Eng. 35 (2021) 04021023, https://doi.org/10.1061/(ASCE)CP.1943-5487.0000988.
[45] P. Chotiprayanakul, D.K. Liu, G. Dissanayake, Human-robot-environment interaction interface for robotic grit-blasting of complex steel bridges, Autom. Constr. 27 (2012) 11–23, https://doi.org/10.1016/j.autcon.2012.04.014.
[46] R.E. Fikes, N.J. Nilsson, Strips: a new approach to the application of theorem proving to problem solving, Artif. Intell. 2 (1971) 189–208, https://doi.org/10.1016/0004-3702(71)90010-5.
[47] Z. Zeng, Z. Zhou, Z. Sui, O.C. Jenkins, Semantic robot programming for goal-directed manipulation in cluttered scenes, in: Proc IEEE Int Conf Robot Autom, Institute of Electrical and Electronics Engineers Inc., 2018, pp. 7462–7469, https://doi.org/10.1109/ICRA.2018.8460538.
[48] B. Hayes, B. Scassellati, Autonomously constructing hierarchical task networks for planning and human-robot collaboration, in: Proc IEEE Int Conf Robot Autom, 2016, pp. 5469–5476, https://doi.org/10.1109/ICRA.2016.7487760.
[49] Z. Zhang, Y. Zhu, S.-C. Zhu, Graph-based Hierarchical Knowledge Representation for Robot Task Transfer from Virtual to Physical World. https://doi.org/10.1109/IROS45743.2020.9340843.
[50] K. Chen, S. Chernova, D. Kent, Learning hierarchical task networks with preferences from unannotated demonstrations. Proceedings of the 2020 Conference on Robot Learning, 2021, pp. 1572–1581, in: https://proceedings.mlr.press/v155/chen21d.html.
[51] B.D. Argall, S. Chernova, M. Veloso, B. Browning, A survey of robot learning from demonstration, Robot. Auton. Syst. 57 (2009) 469–483, https://doi.org/10.1016/J.ROBOT.2008.10.024.
[52] M. Van Otterlo, M. Wiering, Reinforcement learning and Markov decision processes, Adapt. Learn. Optimization 12 (2012) 3–42, https://doi.org/10.1007/978-3-642-27645-3_1.
[53] R. Chitnis, S. Tulsiani, S. Gupta, A. Gupta, Efficient bimanual manipulation using learned task schemas, in: Proc IEEE Int Conf Robot Autom, 2020, pp. 1149–1155, https://doi.org/10.1109/ICRA40945.2020.9196958.
[54] S. Nasiriany, H. Liu, Y. Zhu, Augmenting reinforcement learning with behavior primitives for diverse manipulation tasks, in: IEEE International Conference on Robotics and Automation (ICRA), 2022, https://doi.org/10.48550/arxiv.2110.03655.
[55] A. Singh, H. Liu, G. Zhou, A. Yu, N. Rhinehart, S. Levine, Parrot: data-driven behavioral priors for reinforcement learning, in: International Conference on Learning Representations (ICLR), 2020, pp. 1–19. http://arxiv.org/abs/2011.10024.
[56] K. Pertsch, Y. Lee, J.J. Lim, Accelerating reinforcement learning with learned skill priors, in: 4th Conference on Robot Learning (CoRL), Cambridge MA, USA, 2020. http://arxiv.org/abs/2010.11944.
[57] D. Kent, S. Banerjee, S. Chernova, Learning sequential decision tasks for robot manipulation with abstract markov decision processes and demonstration-guided exploration, in: IEEE-RAS International Conference on Humanoid Robots, 2019, pp. 958–965, https://doi.org/10.1109/HUMANOIDS.2018.8624949.
[58] M.T. Rosenstein, A.G. Barto, Supervised actor-critic reinforcement learning, Learning and Approximate Dynamic Programming: Scaling Up to the Real World. https://doi.org/10.1109/9780470544785.ch14.
[59] H. Dang, P.K. Allen, Robot learning of everyday object manipulations via human demonstration, in: IEEE/RSJ 2010 International Conference on Intelligent Robots and Systems, IROS 2010 - Conference Proceedings, IEEE, 2010, pp. 1284–1289, https://doi.org/10.1109/IROS.2010.5651244.
[60] C.-J. Liang, W. Mcgee, C.C. Menassa, V.R. Kamat, Real-time state synchronization between physical construction robots and process-level digital twins, Construction Robotics (2022), https://doi.org/10.1007/s41693-022-00068-1.
[61] A. Adel, Computational Design for Cooperative Robotic Assembly of nonstandard timber frame buildings, ETH Zurich (2020), https://doi.org/10.3929/ethz-b-000439443.
[62] H. Liu, C. Sydora, M.S. Altaf, S.H. Han, M. Al-Hussein, Towards sustainable construction: BIM-enabled design and planning of roof sheathing installation for prefabricated buildings, J. Clean. Prod. 235 (2019) 1189–1201, https://doi.org/10.1016/j.jclepro.2019.07.055.
[63] R. Strudel, A. Pashevich, I. Kalevatykh, I. Laptev, J. Sivic, C. Schmid, Learning to combine primitive skills: A step towards versatile robotic manipulation, in: Proc IEEE Int Conf Robot Autom, 2020, pp. 4637–4643, https://doi.org/10.1109/ICRA40945.2020.9196619.
[64] A. Paraschos, C. Daniel, J. Peters, G. Neumann, Probabilistic movement primitives, Adv. Neural Inf. Process. Syst. 1–9 (2013). https://dl.acm.org/doi/abs/10.5555/2999792.2999904.
[65] A. Warszawski, D.A. Sangrey, Robotics in building construction, J. Constr. Eng. Manag. 111 (1985) 260–280, https://doi.org/10.1061/(asce)0733-9364(1985)111:3(260).
[66] J.G. Everett. Construction Automation: Basic Task Selection an Development of the Cranium, Massachusetts Institute of Technology, 1991. http://hdl.handle.net/1721.1/13900.
[67] C. Feng, Camera Marker Networks for Pose Estimation and Scene Understanding in Construction Automation and Robotics. https://deepblue.lib.umich.edu/handle/2027.42/113481, 2015.
[68] K.P. Kisi, N. Mani, E.M. Rojas, E.T. Foster, Optimal productivity in labor-intensive construction operations: pilot study, J. Constr. Eng. Manag. 143 (2017) 04016107, https://doi.org/10.1061/(ASCE)CO.1943-7862.0001257.
[69] R.B. Harris. Precedence and Arrow Networking Techniques for Construction, Wiley, 1978, ISBN 978-0-471-04123-8.
[70] S. Chernova, M. Veloso, Confidence-based multi-robot learning from demonstration, Int. J. Soc. Robot. 2 (2010) 195–215, https://doi.org/10.1007/s12369-010-0060-0.
[71] M. Krogius, A. Haggenmiller, E. Olson, Flexible layouts for fiducial tags. https://doi.org/10.1109/IROS40897.2019.8967787.
[72] C.H.G. Li, Y.M. Chang, Automated visual positioning and precision placement of a workpiece using deep learning, Int. J. Adv. Manuf. Technol. 104 (2019) 4527–4538, https://doi.org/10.1007/s00170-019-04293-x.
[73] A. Hornung, K.M. Wurm, M. Bennewitz, C. Stachniss, W. Burgard, OctoMap: an efficient probabilistic 3D mapping framework based on octrees, Auton. Robot. 34 (2013) 189–206, https://doi.org/10.1007/s10514-012-9321-0.
[74] T. Shan, B. Englot, C. Ratti, D. Rus, LIO-SAM: tightly-coupled lidar-visual-inertial odometry via smoothing and mapping, in: Proc IEEE Int Conf Robot Autom, 2021, pp. 5692–5698, https://doi.org/10.1109/ICRA48506.2021.9561996.
[75] ATI, Why ATI Robotic Tool Changers?. https://www.ati-ia.com/company/NewsArticle2.aspx?id=554643677, 2023.
[76] OnRobot, Dual Quick Changer - Improves cycle time and productivity by 50%, 2023. https://onrobot.com/us/products/dual-quick-changer.
[77] I.A. Sucan, M. Moll, L.E. Kavraki, The open motion planning library, IEEE Robot. Autom. Mag. 19 (2012) 72–82, https://doi.org/10.1109/MRA.2012.2205651.
[78] S. Chitta, I. Sucan, S. Cousins, MoveIt!, IEEE Robot. Autom. Mag. 19 (2012) 18–19, https://doi.org/10.1109/MRA.2011.2181749.
[79] McNeel, Rhino - Rhino, Inside (2022). https://www.rhino3d.com/features/rhino-inside/ (accessed March 22, 2022).
[80] A. Zhu, P. Pauwels, B. De Vries, Component-based robot prefabricated construction simulation using IFC-based building information models, Autom. Constr. 152 (2023), 104899, https://doi.org/10.1016/j.autcon.2023.104899.
[81] B. Elkhapery, R. Pěnička, M. Němec, M. Siddiqui, Metaheuristic planner for cooperative multi-agent wall construction with UAVs, Autom. Constr. 152 (2023), https://doi.org/10.1016/j.autcon.2023.104908.
[82] Z. Huang, F. Li, L. Xu, Modeling and simulation of 6 DOF robotic arm based on Gazebo, 2020 6th International Conference on Control, Automation and Robotics, 2020, pp. 319–323, https://doi.org/10.1109/ICCAR49639.2020.9107989.

[83] W. Qian, Z. Xia, J. Xiong, Y. Gan, Y. Guo, S. Weng, H. Deng, Y. Hu, J. Zhang, Manipulation Task Simulation using ROS and Gazebo. https://doi.org/10.1109/ROBIO.2014.7090732.
[84] Y.C. Lin, D. Berenson, Using previous experience for humanoid navigation planning, in: IEEE-RAS International Conference on Humanoid Robots, IEEE Computer Society, 2016, pp. 794–801, https://doi.org/10.1109/HUMANOIDS.2016.7803364.
[85] P.K. Murali, K. Darvish, F. Mastrogiovanni, Deployment and evaluation of a flexible human–robot collaboration model based on AND/OR graphs in a manufacturing environment, Intell. Serv. Robot. 13 (2020) 439–457, https://doi.org/10.1007/S11370-020-00332-9/TABLES/5.
[86] K.M. Lundeen, V.R. Kamat, C.C. Menassa, W. McGee, Autonomous motion planning and task execution in geometrically adaptive robotized construction work, Autom. Constr. 100 (2019) 24–45, https://doi.org/10.1016/j.autcon.2018.12.020.
[87] A. Golabchi, M. Akula, V.R. Kamat, Leveraging BIM for Automated Fault Detection in Operational Buildings. *ISARC*. https://doi.org/10.22260/ISARC2013/0020.
[88] A. Adel, A. Thoma, M. Helmreich, F. Gramazio, M. Kohler, Design of Robotically Fabricated Timber Frame Structures. *38th Annual Conference of the Association for Computer Aided Design in*, Architecture (ACADIA 2018 (2018) 394–403. https://doi.org/10.52842/conf.acadia.2018.394.