Received 18 September 2023, accepted 2 October 2023, date of publication 13 October 2023, date of current version 18 October 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3322933

RESEARCH ARTICLE

Robot Symbolic Motion Planning and Task Execution Based on Mixed Reality Operation

KOKI NAKAMURA AND KOSUKE SEKIYAMA
Department of Mechatronics Engineering, Graduate School of Science and Technology, Meijo University, Nagoya 468-0073, Japan
Corresponding author: Kosuke Sekiyama (sekiyama@meijo-u.ac.jp)

ABSTRACT With the increasing demand for human–robot collaboration (HRC), intuitive interfaces are essential to connect humans and robots. A promising approach is the use of mixed reality (MR) to enhance spatial understanding through virtual and augmented reality. In this paper, we propose a novel HRC system that extracts human handling procedures and generates concrete motion plans for the robot. The user, wearing an MR device, interacts with virtual objects in the MR space using natural hand motions. These motions and resulting state transitions are abstracted into a symbolic semi-order motion planner represented by the reachability graph (RG). Using the RG, an autonomous behavior tree is generated, considering the robot's task environment, and the concrete motion plan is executed by the robot. This system allows the robot to take a more flexible approach to user instructions than conventional MR-HRC systems. Moreover, this system translates human orders into plans that are independent of a specific robot, demonstrating considerable development potential.

INDEX TERMS Human–robot interaction, mixed reality, cyber–physical systems, motion planning, virtual reality.

I. INTRODUCTION

Recently, there has been significant interest in human–robot collaboration (HRC) systems, where humans and robots work together in the same space. An essential requirement to facilitate such collaboration is an interface that enables safe and intuitive communication of information between humans and robots [1].

Although the effectiveness of augmented reality (AR) and virtual reality (VR) as human–robot interfaces in HRC has received considerable attention [2], there are growing expectations for human–robot interaction using mixed reality (MR) [3], MR-HRI technology. MR combines the features of both AR and VR, offering a unique advantage in handling spatial information and seamlessly integrating interactive holographic objects with the real space through spatial computing. Compared with AR, which superimposes virtual objects on real space, and VR, which operates in entirely virtual or reconstructed real environments, MR provides users with an exceptionally immersive experience in the actual space due to its ability to merge virtual and real elements seamlessly [4].

In HRC, the robot requires spatial information about the actual task space and a clear understanding of the work environment for its operations. Traditionally, robots find intuitively conveying their action plans to a person challenging, and real-time operation raises safety concerns. MR-HRI offers a solution by promoting natural and intuitive interactions between humans and robots, as both parties share a common spatial understanding, thus enhancing their interaction capabilities [5], [6]. Utilizing MR objects that can be freely manipulated and expressed in MR space, unconstrained by physical laws, holds the potential to expand functionalities and address challenges in HRC.

In human communication, abstract instructions are frequently conveyed through gestures and directive words; these abstract instructions suggest indirectly the unspoken intentions of the instructor rather than explicitly provide detailed information about the subjects [7], [8]. Moreover,

The associate editor coordinating the review of this manuscript and approving it for publication was Zheng Chen.

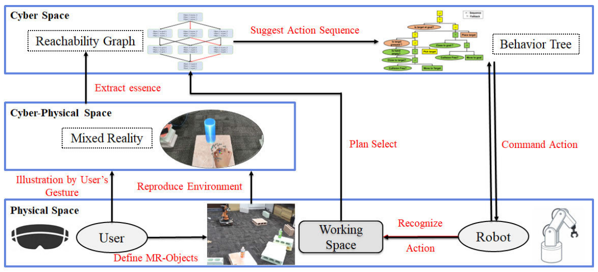


FIGURE 1. Information flow of the SMP-MRO system, which considers three levels of space: Physical Space, Cyber-Physical Space, and Cyber Space. Physical Space is the space where people and robots actually exist, and it includes the work space. Cyber-Physical Space is constructed using MR, where the work space is abstracted and the objects contained in it are given informational meaning. Cyber Space is a complete information space, where user actions and object states are abstracted and exist as information.

many daily life tasks prioritize task completion itself, with less emphasis on the specifics of task execution.

For achieving natural and fluent interaction between human and robots while following abstract human instructions, it is desirable for robots to execute tasks in their own efficient way based on the surrounding task environment.

In this paper, we propose Symbolic Motion Planning based on Mixed Reality Operation (SMP-MRO) as a system that abstracts the operating procedure of task execution demonstrated by object manipulation in MR space into a symbolic partial order plan. The system then reconstructs a motion plan to reach the target state considering the real-space conditions surrounding the robot. In contrast to conventional MR-HRI systems, where robots are taught directly in MR space, SMP-MRO considers the teaching actions indicated by the instructor as viable procedures. After abstraction using a reachability graph (RG), the system generates and executes feasible operation plans, satisfying physical constraints and other conditions in the operating environment, using a behavior tree (BT).

Specifically, SMP-MRO consists of three levels: Physical Space, Cyber-Physical Space, and Cyber Space, as illustrated in Fig.1. In Physical Space, the user associates MR objects with the operation target or robot and defines the MR space, including the task environment, in Cyber-Physical Space. Subsequently, in the MR space, the user intuitively illustrates the action procedure, and data representing the essence of the action intention are sent to Cyber Space. These data correspond to state changes, such as where to place an object in a pick-and-place task. In Cyber Space, the task environment state and actions that cause state transitions are represented by RG. The system generates specific robot action execution instructions using BT and monitors states while selecting actions accordingly.

This approach provides a useful framework for HRC systems, enabling robots to learn and recognize human operating procedures in complex task execution. The effectiveness of SMP-MRO is demonstrated through grasping manipulation experiments with a mobile manipulator.

II. PREVIOUS RESEARCH

A. APPLICATION OF MR-HRI IN HUMAN-ROBOT COLLABORATION

There has been significant research on HRC involving AR/VR/MR technologies. Ostanin et al. represented a system for interactive programming of industrial robots based on MR [4]. They performed the work to planning the EE geometric path for robot intuitively using MR device, and analyzed the advantages of MR compared to AR/VR. This research indicated the added value and potential of MR in intuitive robot programming. Rosen et al. proposed an Action-Oriented Semantic Map (AOSM) that combines object manipulation behaviors and semantic information about the environment to realize complex object manipulation and navigation [9]. They performed the work where the user teaches the robot an AOSM through MR-HMD to plan the robot operation and navigation tasks. Then, they confirmed that object manipulation information and semantic maps are required for high-level action and navigation teaching. These researches contribute to prove the superiority of MR-HRI over conventional systems for unit-of-action programming and teaching of robot in HRC. Compared to these systems, the system presented in this paper allows the autonomous robot to generate a modified motion plan according to the peripheral condition of the robot in order to achieve the intended original plan from the user.

On the other hand, to achieve natural and intuitive interaction in MR-HRC, efficient motion intent communication is required. Rosen et al. proposed a system to visualize proposed robot motion over the user's real-world view of the robot and its environment using MR-HMD [10]. They compared and verified robot arm movement tasks with three different interfaces: no visualization, 2D monitor, and MR. Then, the effectiveness of the MR interface was demonstrated, with a 16% improvement in collision prediction accuracy and a 62% decrease in task completion time compared to the 2D monitor. However, the proposed system only realizes the transmission of motion intentions from the robot to the human, and there remain unresolved issues regarding the transmission from the human to the robot. Maccio et al., carried out a user study to evaluate whether visualization of dynamic robot motion could effectively improve the human-robot collaboration process [11]. The results of the experiment improved the likelihood of collisions and the number of human interventions, but not the time required to complete the task. The reason may be that the superiority of MR devices in HRC cannot be expressed through scripted action plans. To solve these problems, SMP-MRO enables two-way communication of information between humans and robots and creates action sequences that the robots themselves can perform to achieve the illustrated tasks. Taking this into account allows for efficient planning and flexible responses.

We also mention studies that demonstrate the great development potential of MR-HRC systems. Del Merico et al. proposed a system that focuses on spatial computing and egocentric sensing capabilities of MR devices to enable intuitive and natural interactions [12]. They also argued for the potential of the MR-HRC system through various collaborations with robots, such as mission planning for inspection, gesture-based control, and immersive remote control. However, teaching robots about user behavior and making them operate in uncertain environments can be difficult. For these issues, SMP-MRO abstracts the plan through partial-order planning and allows the robot to interpret the plan within its surrounding environment, providing flexibility in the robot's behaviors.

B. RESEARCH ON ACTION PLAN

Classical planning, exemplified by STRIPS [13], is an action plan that assumes complete information and deterministic state transitions. In STRIPS, action sequences are generated based on the action set defined by the preceding state of the action, the result of the action, and the set of initial and target states. For TAMP (task and motion planning), which involves planning the action sequence and generating the robot's movements concurrently, an action plan in a hybrid space that combines continuous variables such as robot or object posture with discrete variables such as grasping state is essential [14]. Garrett et al. proposed the hybrid backward-forward (HBF) algorithm [15]. In HBF, state transitions from the initial state to the target state are performed through a forward search with continuous selection of possible actions at each point in time, leading to the generation of the reachability graph (RG). Backward search utilizes the generated RGs to efficiently sample actions toward the target state. Furthermore, Jial et al. proposed an extended method of cg (contact graph) that incorporates motion relationships detected from the real environment. [16]. This method proves highly effective for tasks that are challenging to accomplish using traditional planning languages, such as Planning Domain Definition Language (PDDL) [17].

In a real environment, comprehending all the necessary environmental information and processing uncertain data, such as sensor recognition and action results, pose significant challenges. Peter et al. proposed the 3T architecture, which involves structuring three hierarchies: the deliberation layer in the upper layer, the sequencing layer in the middle layer, and the reactive skill layer in the lower layer. This architecture enables optimizing different levels of thinking to handle complex tasks effectively [18]. Matsuoka et al. adopted a hierarchical architecture for the robot's error recovery action plan [19]. The recovery process involves setting a new target based on error factor inference using Bayesian networks and modifying and reusing part of the partial order plan.

Recently, there has been a growing trend of applying BTs, commonly used in game AI, to the field of robotics. BTs are utilized for state monitoring and action selection, offering advantages over finite state machines (FSM), particularly its high modularity. French et al. demonstrated the application of BT in achieving high-difficulty robotic learning from demonstration (LfD) tasks, using it as an alternative representation for complex tasks [20]. Their study introduces the BT-Expresso algorithm, which generates a decision tree using CART (classification and regression tree) from illustrations and then converts the generated decision tree into a BT. Colledanchise et al. proposed an algorithm for the automatic generation and updating of BT based on planning algorithms, enabling robot control in dynamic environments [21]. The modularity of BT improves the adaptability of the plan to errors and changes in the environment. Moreover, Colledanchise et al. presented the planning and action using behavior tree (PA-BT) algorithm for the automatic generation of BT [22]. PA-BT expands BT by sampling actions from a template, considering target constraints.

Yang et al. addressed the challenge of operating BT in environments with only partial observation by proposing adjoint sensing and action (ASA), a model that enables the acquisition of expected environmental information. ASA ensures that the operating environment of PA-BT must be completely observable [23]. In ASA, automatic generation and updating of BT are achieved through ASA-BT, an extension of conventional BT, by treating planning problems of robot behavior and sensing as a partially observable Markov decision process (POMDP).

BT can divide the robot's movement and represent it as a continuous sequence of actions. Leveraging this feature, Han et al. enabled the recording of semantic actions by making the robot act based on BT [24], [25].

As demonstrated above, the BT has been used as an execution procedure generated based on abstracted strategic-level action plan such as RG and methods have been proposed to provide flexibility in the robot's action plan based on the current situation. However, there have been no instances of autonomously generating a flexible action plan for a robot from a person's illustrative behavior. To address this issue, this study proposes SMP-MRO, which autonomously generates RG based on a user's illustrative behavior in MR space, achieved through MR-HRI. Subsequently, the robot generates and executes an action plan based on its own physical situation using BT. SMP-MRO combines the advantages of the MR-HRI in the HRC system mentioned earlier and the feasibility of a new HRC system that integrates robot action planning using RG and BT.

III. SYSTEM CONFIGURATION OF SYMBOLIC MOTION PLANNING BASED ON MIXED REALITY OPERATION (SMP-MRO)

A. ILLUSTRATIVE BEHAVIOR IN A MIXED REALITY SPACE

In this study, the MR space is created using Microsoft HoloLens2 as the MR-HMD. The MR system was developed based on the OpenXR standard, utilizing Unity and MRTK (Mixed Reality Toolkit). The user engages in intuitive illustrative actions by manipulating the MR object with hand motions. Additionally, other representations using MR objects can be employed to convey the robot's behavioral intentions to the user.

B. REPRESENTATION OF PARTIAL ORDER PLAN IN REACHABILITY GRAPH

RG is a generation and representation of all reachable state transitions from the initial state to the goal state; it is presented as a partial order plan based on illustrative actions in MR space. When the operation procedure becomes complex, the number of reachable state transition candidates increases exponentially. To address this issue, constraints are taken into account based on the contents of the plan, such as task details, user intent, and physical conditions. This process helps determine whether certain candidates are reachable, narrowing down the options. Following that, a route search using the A* algorithm is performed on the RG, with weights assigned according to the task's specific requirements, such as success rate for tasks requiring precision movements, or movement distance and execution time for tasks achievable with rough motions. The search result is then utilized as the fundamental motion plan for the robot.

C. ROBOT TASK EXECUTION USING BEHAVIOR TREE

Action selection and error recovery processes use PA-BT to enable the robot to execute its action plan within its own surrounding environment in accordance with RG. The states and actions in RG are expressed in a way that is independent of any specific robot, resulting in a highly abstract plan. Hence, the robot needs to compare its potential actions with the surrounding environment and adapt the RG interpretation accordingly. The RG is deployed by acquiring actions capable of performing the necessary state transitions from the action template prepared in advance for each robot.

The RG/BT component operates on a robot operating system (ROS), and the robots operating in this system must be compatible with ROS. Additionally, the MR component is connected to the ROS framework through ROS#.

IV. COMPONENTS OF SYMBOLIC MOTION PLANNING BASED ON MIXED REALITY OPERATION (SMP-MRO)

A. DEFINING THE PLANNING ENVIRONMENT IN THE MIXED REALITY LAYER

1) DEFINITION OF MR SPACE BY LINKING MR-OBJECTS TO REAL OBJECTS

The coordinate system of a robot is typically right-handed. However, in the HoloLens2 process, on the specification of development tools, both the user and robot are assumed to be in a left-handed coordinate system only in MR layer, as shown in Fig.2. In the current task, initial positions ${}^u\boldsymbol{P}_{r_1}, {}^u\boldsymbol{P}_{r_2}, \cdots, {}^u\boldsymbol{P}_{r_n}$ and ${}^u\boldsymbol{P}_{o_1}, {}^u\boldsymbol{P}_{o_2}, \cdots, {}^u\boldsymbol{P}_{o_m}$ of robots $\boldsymbol{r}$ and $\boldsymbol{o}$ in the user's world coordinate system are set for the robot $\boldsymbol{r} = \{r_1, r_2, \cdots, r_n\}$ and objects (bottles) $\boldsymbol{o} = \{o_1, o_2, \cdots, o_m\}$ in the execution environment. The robot's homogeneous transformation matrix for the world coordinate system, based on ${}^u\boldsymbol{P}_{r_n}$, is calculated

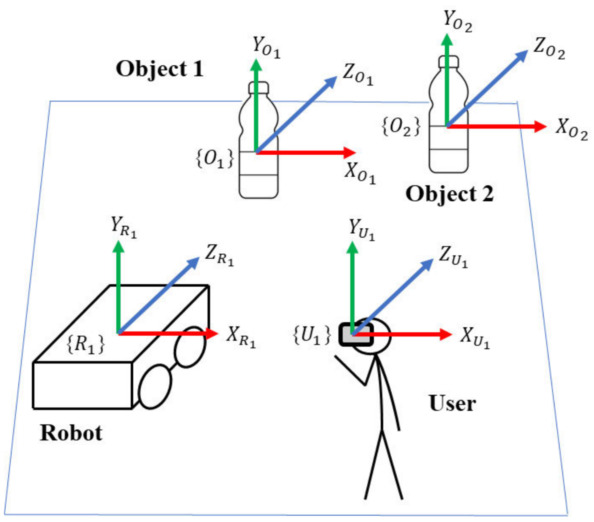


FIGURE 2. Relationship between the coordinate systems of the user, the robot, and each of the objects in the workspace. If the coordinate system orientation and state of each object in the user coordinate system and those in the robot coordinate system differ, the user and robot perceptions of the objects will not match, and therefore coordinate transformation is necessary.

as shown in (1).

$$\boldsymbol{T}^u_{r_n} = \begin{pmatrix} \boldsymbol{R} & -{}^u\boldsymbol{P}_{r_n} \\ 0 & 1 \end{pmatrix} \tag{1}$$

where, $\boldsymbol{P} = (x, y, z)^T \in \boldsymbol{R}^3$ represents the initial state and is expressed in 3-dimensional coordinates, and $\boldsymbol{R}$ represents the rotation matrix in the left-hand coordinate system of the user coordinate system. However, in this paper, we do not acquire information to calculate the rotation matrix as we proceed under the assumption that the world coordinate systems of the user and the robot are aligned in the same direction.

Using the homogeneous transformation matrix, we calculate the initial position of the object to be manipulated in the world coordinate system of each robot, as shown in (2).

$$\begin{pmatrix} {}^{r_n}\boldsymbol{P}_o \\ 1 \end{pmatrix} = \boldsymbol{T}^u_{r_n} \begin{pmatrix} {}^u\boldsymbol{P}_o \\ 1 \end{pmatrix} \tag{2}$$

Through the aforementioned process, we obtain the initial positions ${}^{r_n}\boldsymbol{P}_{o_1}, {}^{r_n}\boldsymbol{P}_{o_2}, \cdots, {}^{r_n}\boldsymbol{P}_{o_m}$ of the object in the world coordinate system of each robot.

2) TARGET STATE SETTING BY ILLUSTRATIVE BEHAVIOR IN MR SPACE

By manipulating the object $\boldsymbol{o}$, the target state or transition points, denoted as ${}^u\boldsymbol{G} = \{{}^u\boldsymbol{G}_1^{\text{id}}, {}^u\boldsymbol{G}_2^{\text{id}}, \cdots, {}^u\boldsymbol{G}_l^{\text{id}}\}$ are set. Here, $\boldsymbol{G} = (x, y, z)^T \in \boldsymbol{R}^3$ represents the target state in 3D coordinates. The association between the generated target state ${}^u\boldsymbol{G}$ and the object's ID: $id \in \{1, 2, \cdots, m\}$ is established. If there are order specifications for the operations, the constraint condition is expressed as $\boldsymbol{G}_i \leq \boldsymbol{G}_j$. Using the homogeneous transformation matrix calculated when defining the MR space, ${}^u\boldsymbol{G}$ is transformed into the world coordinate system of each robot, as depicted in (3).

$$\begin{pmatrix} {}^{r_n}\boldsymbol{G} \\ 1 \end{pmatrix} = \boldsymbol{T}_{r_n}^{u} \begin{pmatrix} {}^{u}\boldsymbol{G} \\ 1 \end{pmatrix} \tag{3}$$

Using the above process, we obtain the target state ${}^{r_n}\boldsymbol{G}_n^{\mathrm{id}} = \{{}^{r_n}\boldsymbol{G}_1^{\mathrm{id}} r_n\boldsymbol{G}_2^{\mathrm{id}}, \cdots, {}^{r_n}\boldsymbol{G}_l^{\mathrm{id}}\}$ and the constraint condition $\boldsymbol{C} = \{\boldsymbol{C}_1, \boldsymbol{C}_2, \cdots, \boldsymbol{C}_k\}$ of the object in the world coordinate system of each robot.

B. ABSTRACTION AND DETERMINATION OF PLANS AT THE REACHABILITY GRAPH LAYER

Alg.1 outlines the process of generating RG based on the information received from the MR layer.

First, receive ${}^{r_n}\boldsymbol{P}_o, {}^{r_n}\boldsymbol{G}^{id}$, and $\boldsymbol{C}$ from the MR layer. However, as ${}^{r_n}\boldsymbol{P}_o$ and ${}^{r_n}\boldsymbol{G}^{id}$ are defined in the left-handed coordinate system, they need to be transformed to the robot's world coordinate system (right-handed coordinate system) by multiplying them with the matrix $\boldsymbol{A}$, as shown in (4), and (5). The appropriate matrix $\boldsymbol{A}$ is set for each robot.

$${}^{r}\boldsymbol{P_o} = \boldsymbol{A}^{r_n}\boldsymbol{P_o} \tag{4}$$

$${}^{r}\boldsymbol{G}^{id} = \boldsymbol{A}^{r_n}\boldsymbol{G}^{id} \tag{5}$$

Generate RG from the calculated ${}^{r}\boldsymbol{P_o}$ and ${}^{r}\boldsymbol{G}^{id}$, and the received $\boldsymbol{C}$.

1) CREATION OF CONDITION NODE

In Alg.1, the GenerateFirstNode function initially creates a node representing the initial state of object $\boldsymbol{o}$. The EXPANDNGRAPH function sequentially creates nodes representing transitionable states from two lists: P, which contains states of object $\boldsymbol{o}$, and G, which contains target states. P' is obtained by converting P using the ChangeState function to represent the post-transition state. Additionally, DeleteGoal is used to remove the achieved goal states from G, resulting in a list of unachieved goal states denoted as G'. Based on P' and G', RG is further extended depth-first by employing the EXPANDGRAPH function.

Algorithm 1 Generate Reachability Graph

```
1:  Input:
2:      $P \leftarrow \{{}^{r}\boldsymbol{P}_{o_1}, {}^{r}\boldsymbol{P}_{o_2}, \cdots, {}^{r}\boldsymbol{P}_{o_m}\}$
3:      $G \leftarrow \{{}^{r}\boldsymbol{G}_1^{id}, {}^{r}\boldsymbol{G}_2^{id}, \cdots, {}^{r}\boldsymbol{G}_l^{id}\}$
4:      constraints
5:  Output:
6:      $RG$(ReachabilityGraph)
7:  $RG$.GenerateFirstNode($P$, $G$)
8:  function ExpandGraph($P$, $G$)
9:      for all $goal \in G$ do
10:         if $goal$ not in constraints then
11:             $P' \leftarrow$ ChangeState($P$, $goal$)
12:             $G' \leftarrow$ DeleteGoal($G$, $goal$)
13:             $RG$.GenerateNode($P'$)
14:             if $G' \neq \{\phi\}$ then
15:                 EXPANDEGRAPH($P'$, $G'$)
16:     return $RG$
```

2) SEARCHING FOR A SEQUENCE OF ACTION

A plan search is conducted by assigning arbitrary weights to the edges of the created RG. For example, in a sorting task, the weight can be set based on the expected travel distance of a robot to pick and place an object, enabling a more efficient achievement of the goal compared to blindly following instructions. The plan resulting from the search is represented as the goal constraint C_{goal} and sent to the BT layer.

C. MOTION PLANNING AT THE BEHAVIOR TREE LAYER

In BT, nodes are ticked at regular intervals, and the constituent nodes are executed in a depth-first manner. The main components of BT are the Control node, Condition node, and Action node. The explanation of each node is as follows.

Control Node: Within the Control Node, there are the Sequence Node ($\rightarrow$) that executes its child nodes in order and fails itself if even one child node fails, and the Fallback Node (?) that fails itself if all child nodes fail. When executing its own child nodes in order, there are a Sequence Node($\rightarrow$) that fails itself if even one child node fails, and a Fallback Node(?) that fails itself if all child nodes fail exist in the Control Node. There are also Reactive Sequence Node (${}^{r} \rightarrow$) and Reactive Fallback Node (${}^{r}?$) that redo the execution of child nodes each time they are ticked, enabling action planning that includes re-actions. Additionally, the Priority Fallback Node (${}^{p}?$) allows plan switching during BT execution. It cancels and switches running child nodes if there is a change in their priority.

1) FORMULATION OF ACTION PLAN

The Configuration Space (C-Space)S is the Cartesian product of the set of variables $V_1, \ldots, V_n$ that constitute the C-Space. A state $s = (v_1, \ldots, v_n) \in S$ consists of values v_i assigned to each variable V_i. The constraint conditions $C^1, \ldots, C^K$ are expressed as conditional expressions involving one or more state variables V_i, which evaluate to either True or False. Actions a comprise a set $a.con = \{C_1, \ldots, C_K\}$ representing means preconditions and a set $a.eff = \{E_1, \ldots, E_l\}$ representing effects. Both $a.con$ and $a.eff$ are sets of constraints. Let $a.eef(s)$ denote the effect of an action in state $s = (v_1, \ldots, v_n)$ that satisfies precondition $a.con$. $a.eef(s)$ updates the elements in state s to reflect the effect of the action.

The planning problem is to find an action sequence $a_1, \ldots, a_m$ that satisfies $a_m.eef(\ldots a_1 s_0)) \in C_{goal}$, given a set Γ of allowable actions, starting from the initial state s_0 and reaching the goal state.

2) GENERATING THE INITIAL TREE

Alg.2 represents the main loop of BT generation. To fulfill the received C_{goal} from the RG layer, SequenceNode generates a Sequence Node that includes a Condition Node representing the target constraint as a child node. If the node fails, GetFailedNode identifies the Condition Node n_f that failed in the breadth-first search.

Algorithm 2 Reactive Execution with Behavior Tree

```
1:  $T \leftarrow \emptyset$
2:  for $c$ in $C_{goal}$ do
3:      $T \leftarrow$ SequenceNode($T$, $c$)
4:  while True do
5:      do
6:          $r \leftarrow$ Tick($T$)
7:      while $r \neq$ Failure
8:      $n_f \leftarrow$ GetFailedNode($T$)
9:      if $n_f$ is ConditionNode then
10:         $T \leftarrow$ EXPANDTREE($T$, $n_f$)
```

3) CREATION OF SUB TREE

The expansion of T by ExpandNode is carried out as shown in Alg.3. First, GetAllActTemplatesFor searches for actions from the action templates that satisfy the failed Condition Node c_f. Unlike PA-BT, all actions with the effect of satisfying satisfying c_f are generated, making the priority of actions variable. Next, as a child node of ReactiveSequence, a precondition Condition node c_a and an Action node are created in that order. Finally, the algorithm checks for any constraint conflicts. When conflicts occur, it handles them by increasing the priority of the added subtree until the conflicts are resolved. If the subtree has the highest priority, it will be generated above the priority of its parent.

V. FUNCTIONAL VERIFICATION THROUGH ACTUAL EXPERIMENTS

The function of this system was verified by conducting three patterns of operation confirmation experiments involving the Pick-Place task.

A. DEFINITION OF STATES AND ACTIONS HANDLED BY YOUBOT

In this experiment, the mobile manipulator youBot (KUKA) is used as the robot to be instructed. YouBot does not have a pre-defined map, but it constructs the map online during operation. The origin of the world coordinate system of the generated map aligns with the origin of the base coordinate system when youBot is activated. The definition of states and actions depends on the content of the task, and for youBot, the states and actions are defined as follows.

1) DEFINITION OF STRUCTURES AND STATE VARIABLES

The structure handled by the state variable is defined as follows.

- **Point**: stores position in 3 dimensions (x, y, z):
  double x, double y, double z
- **Quaternion**: quaternion:
  double x, double y, double z, double w
- **Pose**: positional posture expression:
  Point *pos*, Quaternion *orientation*

Algorithm 3 Behavior Tree Expansion

```
1:  function ExpandTree($T$, $c_f$)
2:      $A_T \leftarrow$ GetAllActTemplatesFor($c_f$)
3:      if $A_T$ is $\emptyset$ then
4:          $T \leftarrow$ GetRecoveryTreeForCondition($T$, $c_f$)
5:          return $T$
6:      $T_{rfall} \leftarrow$ ReactiveFallbackNode($c_f$)
7:      $T_{pfall} \leftarrow \emptyset$
8:      for $a$ in $A_T$ do
9:          $T_{seq} \leftarrow \emptyset$
10:         $a \leftarrow$ GetActionValuesFromCondition($a$, $c_f$)
11:         for $c_a$ in $a.con$ do
12:             $a \leftarrow$ GetConditionValuesFromAction($c_a$, $a$)
13:             $T_{seq} \leftarrow$ ReactiveSequenceNode($T_{seq}$, $c_a$)
14:             if $c_a$ is not satisfied then
15:                 $T_{seq} \leftarrow$ EXPANDTREE($T_{seq}$, $c_a$)
16:         $T_{seq} \leftarrow$ ReactiveSequenceNode($T_{seq}$, $a$)
17:         $T_{pfall} \leftarrow$ PriorityFallbackNode($T_{pfall}$, $T_{seq}$)
18:     $T_{rfall} \leftarrow$ ReactiveFallbackNode($T_{rfall}$, $T_{pfall}$)
19:     $T \leftarrow$ Replace($T$, $c_f$, $T_{rfall}$)
20:     while Conflict($T$) do
21:         $T \leftarrow$ IncreasePriority($T_{rfall}$)
22:     return $T$
```

- **Area**: expression of positional area (circular with radius specified):
  Point *pos*, double *radius*
- **Object**: expression of object information (uid is the id for identification):
  string *uid*, Pose *pose*

However, Pose represents the map coordinate system that youBot has.

For Area l, we denote $p \in l$ when Point p is in the domain.

Regarding the set of variables $V_1, \ldots, V_n$ constituting the C-Space, the state variables handled by youBot in the Pick-Place action plan for this experiment are defined as follows.

- Pose r: self-positioning of youBot
- string h: uid. of the grasped object. If it is not grasped, $h =$ None
- Object o_{uid}: known object info
- Area l_{uid}: given position range

2) CONSTRAINTS TEMPLATE

Constraints can be True or False. youBot handles constraint conditions as follows.

- **IsObjectAt**(Object o, Area l):
  True for $o.pose.pos \in l$, or False otherwise
- **IsRobotCloseTo**(Point p, double $d = 0.9$):
  True for $r.pos \in l$ for a circular area l with position p and radius d, or False otherwise
- **IsHandFree**():
  True for $h =$ None, or False otherwise
- **IsGraspdObject**(string *uid*):
  True for $h = uid$, or False otherwise
- **IsAreaFree**(Area l):
  True for $\forall o_{uid}.pose.pos \notin l$, or False otherwise

3) ACTION TEMPLATE

An action template defines the actions that youBot can perform. It consists of a precondition (con) and an effect (eff). An action can be executed if the preconditions are met. Successful execution of the action satisfies the constraint conditions set for the effect. In this experiment, **MoveTo**, **Pick**, and **Place** are defined as follows.

MoveTo(Area l):
con:
eff: IsRobotCloseTo($l.pos$, $l.radius$)

Pick(Object o):
con: IsHandFree()
IsRobotCloseTo($o.pos$)
eff: IsGraspdObject($o.uid$)

Place(Object o, Area l):
con: IsGraspdObject($o.uid$)
IsRobotCloseTo($l.pos$, $l.radius$)
IsAreaFree(l)
eff: IsHandFree()
IsObjectAt(o, l)

B. CASE1: SIMPLE PICK-PLACE TASK

In this task, we checks to see the robot select an efficient approach in own surrounding environment and complete the plan without simply following the plan as indicated by the user.

1) PLANNING IN MIXED REALITY SPACE

The Case1 experiment is performed in the environment shown in Fig.3a. Users give natural instructions by planning for objects located close to themselves. In the environment of Fig.3a, the user plans are following order:

$$\begin{array}{ll}(\text{U1}) & O_0 \in P_0 \rightarrow O_0 \in G_0\\ (\text{U2}) & O_1 \in P_1 \rightarrow O_1 \in G_1\\ (\text{U3}) & O_2 \in P_2 \rightarrow O_2 \in G_2\end{array}$$

After planning, the actual MR space becomes Fig.3b.

2) PROPOSED SEQUENCE OF ACTIONS BY GENERATING REACHABILITY GRAPH

The information obtained from planning is $\boldsymbol{P} = \{P_0, P_1, P_2\}$, $\boldsymbol{G} = \{G_0, G_1, G_2\}$. An RG of Fig.4 was generated for this information. In this RG, nodes $O_0 \in P_0, O_1 \in P_1, O_2 \in P_2$ were initially created to represent the initial state. When any node is generated, the RG grows by generating new nodes for target states that are not satisfied at that node and also do not satisfy the constraints one by one, in depth-first order. In this way, all reachable state transitions can be represented, and all edges terminate at nodes that finally satisfy all target states ($O_0 \in G_0, O_1 \in G_1, O_2 \in G_2$). The above generation confirms that all reachable state transitions are represented in the generation of RG.

3) DEPLOYMENT OF RG TO BT BY ROBOT ACTION FOR AUTONOMOUS TASK EXECUTION

An initial tree of BT is generated for the selected sequence of state transitions. In this experiment, an initial tree was generated under the Sequence node with additional state nodes in the order IsObjectAt(O_2, l_2), IsObjectAt(O_1, l_1), IsObjectAt(O_0, l_0). L_0, L_1, and L_2 represent the target regions centered on G_0, G_1, and G_2, respectively. When the robot actually acts, it ticks the initial tree in sequence, but the initial tree needs to grow because all objects are in their initial state and do not meet the target state. As an example, the BT grown to satisfy IsObjectAt(O_2, l_2) is shown in Fig.5. By performing BT growth for these unsatisfied state and action nodes, the robot performs the task autonomously. For IsObjectAt(O_1, l_1) and IsObjectAt(O_0, l_0), BT grew as before.

4) CONFIRMATION OF ACTUAL OPERATION

YouBot executed the task autonomously as shown in Fig.6 through the planning in MR, RG generation, and BT growth. In response to the user's instruction for the following procedure:

$$\begin{array}{ll}(\text{U1}) & O_0 \in P_0 \rightarrow O_0 \in G_0\\ (\text{U2}) & O_1 \in P_1 \rightarrow O_1 \in G_1\\ (\text{U3}) & O_2 \in P_2 \rightarrow O_2 \in G_2\end{array}$$

youBot executed the operations in the following order:

$$\begin{array}{ll}(\text{R1}) & O_2 \in P_2 \rightarrow O_2 \in G_2\\ (\text{R2}) & O_1 \in P_1 \rightarrow O_1 \in G_1\\ (\text{R3}) & O_0 \in P_0 \rightarrow O_0 \in G_0\end{array}$$

From this experiment, it was confirmed that this system can select the most appropriate approach for the robot's surrounding environment, taking the user's teaching into account.

C. CASE2: PICK-PLACE TASK WITH CONSTRAINTS

In this experiment, we will verify the generation of appropriate RGs according to the constraints by performing a Pick-Place task involving reordering where constraints occur.

1) PLANNING IN MIXED REALITY SPACE

The Case2 experiment is conducted in the environment shown in Fig.7a. The user performs a swapping action by placing O_0 and O_1 in the free space O_0 is placed where O_1 was, and O_1 is placed where O_0 was. The environment after planning is shown in Fig.7b. The user plans in the following order:

$$\begin{array}{ll}(\text{U1}) & O_0 \in P_0 \rightarrow O_0 \in G_0\\ (\text{U2}) & O_1 \in P_1 \rightarrow O_1 \in G_1\\ (\text{U3}) & O_0 \in G_0 \rightarrow O_0 \in G_2\\ (\text{U4}) & O_1 \in G_1 \rightarrow O_1 \in G_3\end{array}$$

In this task, O_0 and O_1 are moved twice each, so constraint conditions are needed to represent the order of operations according to the user's intention. The generated constraints to express the intention are $G_0 \leq G_2$ and $G_1 \leq G_3$. Additionally, physical constraints are required to ensure that objects do not exist at the placement destination during reordering, so $G_0 \leq G_3$ and $G_1 \leq G_2$ are generated.

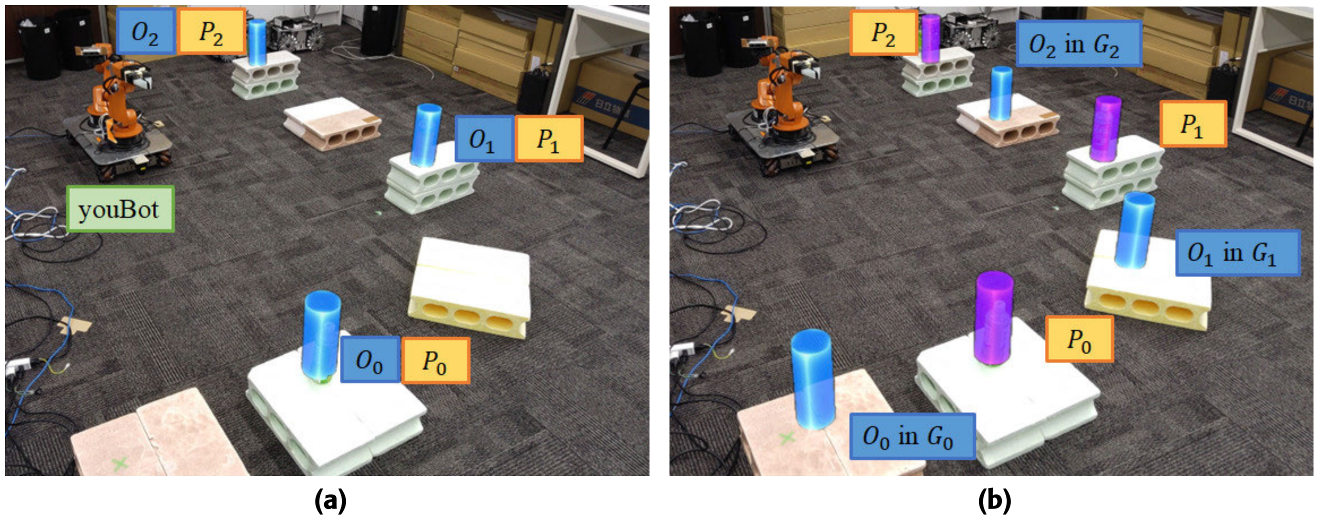


FIGURE 3. (a) : Reconstruction of the work environment of the Case 1 experiment in MR space, as seen from the user's perspective. The blue MR objects are tied to objects in the real environment. (b): Target state set by the user's illustrative behavior to the environment in (a). The blue MR object in (b) represents the final target state of the object defined in (a), and the purple MR object represents the initial or intermediate target state of one of the MR objects.

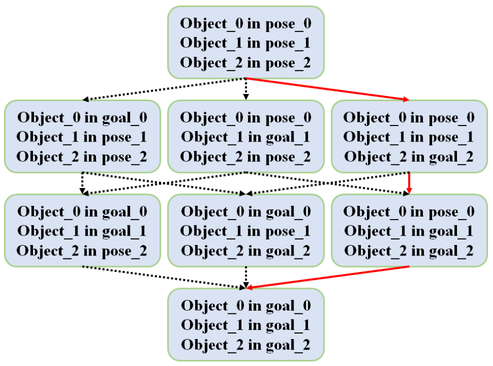


FIGURE 4. RG generated for the target state set in Fig.3b. Each node represents a state in the workspace, and the arrow edges of the dotted line represent state transitions. Additionally, the arrow edges of the practice are the sequence of state transitions selected considering the robot's surrounding environment.

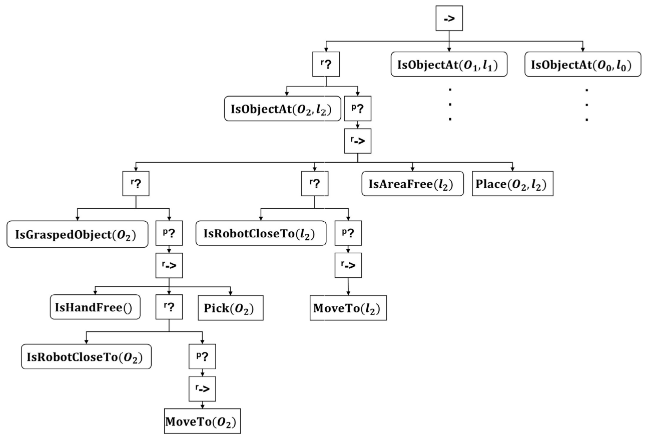


FIGURE 5. Sequence of state transitions chosen in Fig.4 is executed as the robot grows BT in its own way. This figure shows the BT grown from the initial tree to execute the state battle in $O_2 \in P_2 \rightarrow O_2 \in G_2$.

2) GENERATION OF REACHABILITY GRAPH CONSIDERING CONSTRAINTS

The information obtained from planning is $\boldsymbol{P} = \{P_0, P_1\}$, $\boldsymbol{G}$=$\{G_0, G_1, G_2, G_3\}$, $\boldsymbol{C}$=$\{G_0{\le}G_2, G_1{\le}G_3, G_0{\le}G_3, G_1{\le}G_2\}$. RG of Fig.8 was generated for this information. In this RG, a node representing the initial state ($O_0 \in P_0, O_1 \in P_1$) was generated, and as in Case 1, it attempted to generate a node with state transitions to the unachieved target state $\{G_0, G_1, G_2, G_3\}$. However, none of the constraints $\{G_0 \le G_2, G_1 \le G_3, G_0 \le G_3, G_1 \le G_2\}$ are satisfied, so the only target state that can make a state transition is $\{G_0, G_1\}$. Then, when all edges converge at the node ($O_0 \in G_0, O_1 \in G_1$) representing the state of moving all objects into the free space, RG is generated for the remaining target states $\{G_2, G_3\}$ because all the constraints are satisfied at this node, and RG is completed. Finally, as in Case 1, a sequence of state transitions is selected.

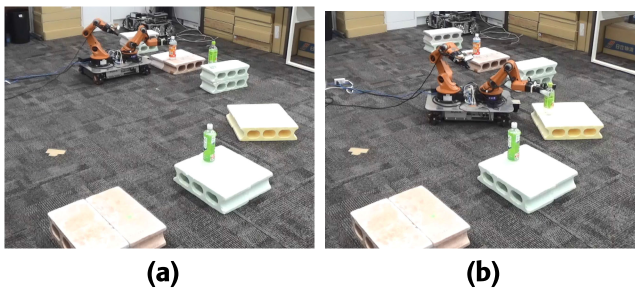


FIGURE 6. Robot in action to carry out the user's instructions. (a) : Execution of $O_2 \in P_2 \rightarrow O_2 \in G_2$, (b): Execution of $O_1 \in P_1 \rightarrow O_1 \in G_1$ and $O_0 \in P_0 \rightarrow O_0 \in G_0$.

3) CONFIRMATION OF ACTUAL OPERATION

YouBot executed the task autonomously as shown in Fig.9 through the planning in MR, RG generation, and BT growth

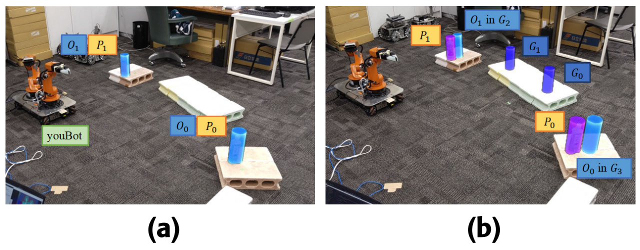


FIGURE 7. (a) : Reconstruction of the work environment of the Case 2 experiment in MR space, as observed from the user's perspective. The blue MR objects are tied to objects in the real environment. (b): Target state set by the user's illustrative behavior to the environment in (a). The blue MR object in (b) represents the final target state of the object defined in (a), and the purple MR object represents the initial or intermediate target state of one of the MR objects.

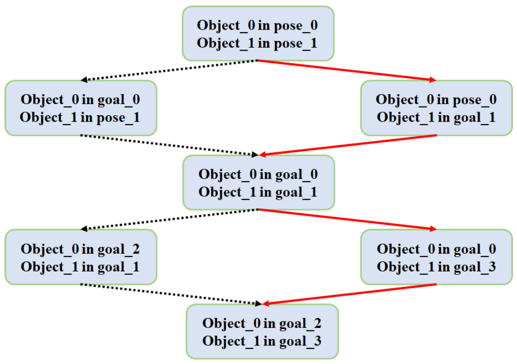


FIGURE 8. RG generated for the target state set in Fig.7b. Fewer state transitions are actually generated compared to the state transitions assumed from the length of the user's movement sequence. This is because state transitions that do not satisfy the constraint conditions were eliminated.

process. In response to the user's instruction for the following procedure:

$$\begin{aligned}
&(\mathrm{U1})\quad O_0 \in P_0 \rightarrow O_0 \in G_0\\
&(\mathrm{U2})\quad O_1 \in P_1 \rightarrow O_1 \in G_1\\
&(\mathrm{U3})\quad O_0 \in G_0 \rightarrow O_0 \in G_2\\
&(\mathrm{U4})\quad O_1 \in G_1 \rightarrow O_1 \in G_3
\end{aligned}$$

youBot executed the operations in the following order:

$$\begin{aligned}
&(\mathrm{R1})\quad O_1 \in P_1 \rightarrow O_1 \in G_1\\
&(\mathrm{R2})\quad O_0 \in P_0 \rightarrow O_0 \in G_0\\
&(\mathrm{R3})\quad O_1 \in G_1 \rightarrow O_1 \in G_3\\
&(\mathrm{R4})\quad O_0 \in G_0 \rightarrow O_0 \in G_2
\end{aligned}$$

The formation and growth of BTs also took place in this case as in Case 1. This experiment confirmed that RG generation correctly enumerates only reachable state transitions according to the constraints.

D. CASE3: ROBOT AUTONOMOUS RECOVERY TO PERFORM TASK

In this experiment, obstacles that were not defined in the planning environment were added during task execution. Under this condition, we check if the robot is capable of autonomous plan execution while making a recovery plan.

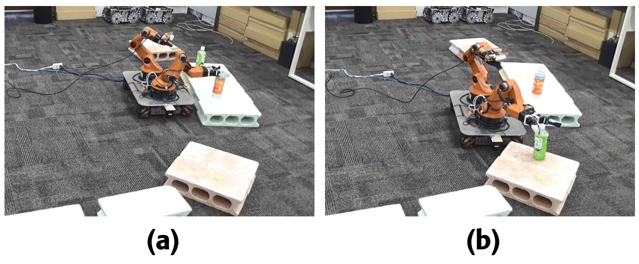


FIGURE 9. Robot in action to carry out the user's instructions. (a) :Execution of $O_1 \in P_1 \rightarrow O_1 \in G_1$ and $O_0 \in P_0 \rightarrow O_0 \in G_0$, (b):Execution of $O_1 \in G_1 \rightarrow O_1 \in G_3$ and $O_0 \in G_0 \rightarrow O_0 \in G_2$.

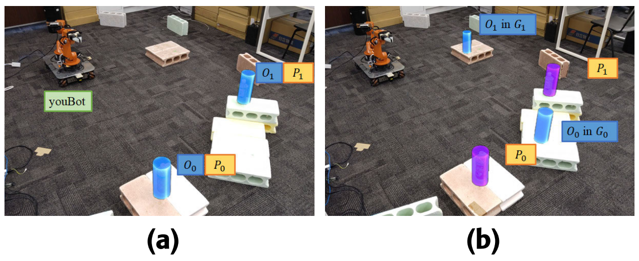


FIGURE 10. (a) : Reconstruction of the work environment of the Case 3 experiment in MR space, as observed from the user's perspective. The blue MR objects are tied to objects in the real environment. (b): Target state set by the user's illustrative behavior to the environment in (a). The blue MR object in (b) represents the final target state of the object defined in (a), and the purple MR object represents the initial or intermediate target state of one of the MR objects.

1) PLANNING IN MIXED REALITY SPACE

The Case3 experiment is performed in the environment shown in Fig.10a. So as to move each object, the user planed following order:

$$\begin{aligned}
&(\mathrm{U1})\quad O_0 \in P_0 \rightarrow O_0 \in G_0\\
&(\mathrm{U2})\quad O_1 \in P_1 \rightarrow O_1 \in G_1
\end{aligned}$$

The target state resulting from this planning is shown in Fig.10b.

2) RECOVERY SUPPORT BY BEHAVIOR TREE GROWTH

The robot performs autonomous task execution by generating RG for the information obtained from planning, selecting a sequence of state transitions, and materializing them into BT. In the current experiment, however, an obstacle O_a, which did not exist when the planning environment was defined, is placed at the location of G_1 during task execution. As in Case 1, YouBot aims to perform the task by growing BT, but when executing IsObjectAt(O_1, l_1) to satisfy $O_1 \in P_1 \rightarrow O_1 \in G_1$, the Condition Node of IsAreaFree(l_1) is not satisfied, and the Action Node of Place(O_1, l_1) fails, which forces the sub-tree to grow. Therefore, a tree was added to check the conditions of IsObjectAt(O_a, l_{free1}) to satisfy IsAreaFree(l_1), as shown in Fig.11. l_{free} refers to any free space that does not affect task execution. In this case, the Condition Nodes of IsGraspedObject(O_1) and IsGraspedObject(O_a) cannot be satisfied simultaneously, and interference of constraint conditions occurs. Hence, the sub-tree was moved up in the order of execution until the

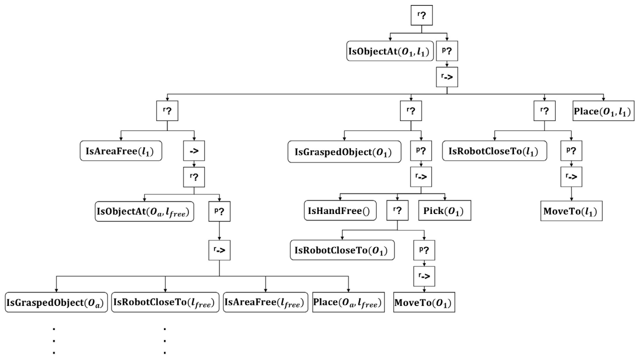


FIGURE 11. **BT with an additional recovery tree that removes obstructions to place O_1. The addition of the recovery tree enables flexibility while dealing with unforeseen conditions.**

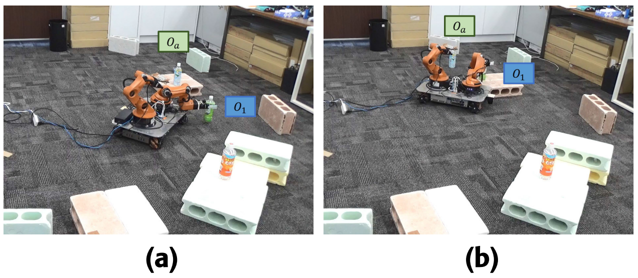


FIGURE 12. **Robot actually in action to carry out the user's instructions. (a) :Discovery of O_a and placement of O_1 into l_{free}, (b):O_a Elimination of O_a and rearrangement of O_1.**

interference was resolved. Since YouBot is grasping O_1 at this stage, Place(O_1, l_{free2}) is performed to eliminate O_a, and O_a is recovered by placing it on l_{free1}.

3) CONFIRMATION OF ACTUAL OPERATION

YouBot executed the task autonomously as shown in Fig.12 through the planning in MR, RG generation, and BT growth. As in Cases 1 and 2, RGs were generated from the user's illustrated plan. Additionally, in the autonomous task execution using BT, for obstacle O_a, YouBot eliminated O_a to free space l_{free1} by placing O_1 once in the free area l_{free2}. Subsequently, it acted again to satisfy $O_1 \in G_1$. This confirms that the robot is capable of planning its own recovery plan and executing it autonomously, even for conditions that are not defined in the planning environment.

VI. CONCLUSION

In this paper, we proposed a system that includes intuitive planning by MR, abstracting the plan by RG, and searching for optimal approaches for arbitrary indicators, such as work time and travel distance, to enable autonomous task execution of robots using BT, which resembles instructions from a person to a robot. Furthermore, we conducted experiments to verify the feasibility and effectiveness of this system. The adaptability of the system can be achieved by constructing an appropriate MR space for the environment and preparing an action template suitable for the specific robot.

SMP-MRO is the system that realizes smooth and natural communication between humans and robots. Therefore, this system can be expected to be used in situations where robots that can flexibly interpret and communicate human instructions are required, such as coexistence with robots in daily life, education, and welfare. However, there are still major issues that need to be improved in order to put SMP-MRO into practical use, as shown below.

In the MR layer, automating the definition of planning environment and increasing the means of communication with robots are important issues to reduce the load on users. The RG layer should allow SMP-MRO to automatically select weights based on the task content to accommodate more general situations. Furthermore, the system can be further optimized by re-proposing options to the robot. The BT layer provides more flexibility by automating the selection process based on evaluations such as recovery plan success rate and execution time, and deciding whether to fix a failed recovery plan or start another recovery plan. This leads to efficient task execution.

As mentioned above, these improvements hold great promise for further development and sophistication of the system, and the practical application of SMP-MRO is a future challenge.

REFERENCES

[1] S. Li, P. Zheng, S. Liu, Z. Wang, X. V. Wang, L. Zheng, and L. Wang, "Proactive human–robot collaboration: Mutual-cognitive, predictable, and self-organising perspectives," *Robot. Computer-Integrated Manuf.*, vol. 81, Jun. 2023, Art. no. 102510.
[2] D. Whitney, E. Rosen, E. Phillips, G. Konidaris, and S. Tellex, "Comparing robot grasping teleoperation across desktop and virtual reality with ROS reality," in *Robotics Research*, 2019, pp. 335–350.
[3] P. Milgram and F. Kishino, "A taxonomy of mixed reality visual displays," *IEICE Trans. Inf. Syst.*, vols. E77–D, no. 12, pp. 1321–1329, 1994.
[4] M. Ostanin and A. Klimchik, "Interactive robot programing using mixed reality," *IFAC-PapersOnLine*, vol. 51, no. 22, pp. 50–55, 2018.
[5] A. Demian, M. Ostanin, and A. Klimchik, "Dynamic object grasping in human-robot cooperation based on mixed-reality," in *Proc. Int. Conf. Nonlinearity, Inf. Robotics (NIR)*, Aug. 2021, pp. 1–5.
[6] M. Khatib, K. Al Khudir, and A. De Luca, "Human-robot contactless collaboration with mixed reality interface," *Robot. Comput.-Integr. Manuf.*, vol. 67, Feb. 2021, Art. no. 102030.
[7] B. Mutlu, F. Yamaoka, T. Kanda, H. Ishiguro, and N. Hagita, "Nonverbal leakage in robots: Communication of intentions through seemingly unintentional behavior," in *Proc. 4th ACM/IEEE Int. Conf. Hum. Robot Interact.*, 2009, pp. 69–76.
[8] E. Calisgan, A. Haddadi, H. F. M. Van der Loos, J. A. Alcazar, and E. A. Croft, "Identifying nonverbal cues for automated human-robot turn-taking," in *Proc. IEEE RO-MAN: 21st IEEE Int. Symp. Robot Hum. Interact. Commun.*, Sep. 2012, pp. 418–423.
[9] E. Rosen, N. Kumar, N. Gopalan, D. Ullman, G. Konidaris, and S. Tellex, "Building plannable representations with mixed reality," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2020, pp. 11146–11153.
[10] E. Rosen, D. Whitney, E. Phillips, G. Chien, J. Tompkin, G. Konidaris, and S. Tellex, "Communicating robot arm motion intent through mixed reality head-mounted displays," in *Robotics Research*, 2020, pp. 301–316.
[11] S. Maccio, A. Carfí, and F. Mastrogiovanni, "Mixed reality as communication medium for human-robot collaboration," in *Proc. Int. Conf. Robot. Autom. (ICRA)*, May 2022, pp. 2796–2802.
[12] J. Delmerico, R. Poranne, F. Bogo, H. Oleynikova, E. Vollenweider, S. Coros, J. Nieto, and M. Pollefeys, "Spatial computing and intuitive interaction: Bringing mixed reality and robotics together," *IEEE Robot. Autom. Mag.*, vol. 29, no. 1, pp. 45–57, Mar. 2022.

[13] R. E. Fikes and N. J. Nilsson, ''Strips: A new approach to the application of theorem proving to problem solving,'' *Artif. Intell.*, vol. 2, nos. 3–4, pp. 189–208, Dec. 1971.

[14] C. R. Garrett, ''Integrated task and motion planning,'' *Annu. Rev. Control Robot. Auton. Syst.*, vol. 4, pp. 265–293, May 2021.

[15] C. R. Garrett, T. Lozano-Pérez, and L. P. Kaelbling, ''Backward-forward search for manipulation planning,'' in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Sep. 2015, pp. 6366–6373.

[16] Z. Jiao, Y. Niu, Z. Zhang, S.-C. Zhu, Y. Zhu, and H. Liu, ''Sequential manipulation planning on scene graph,'' in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2022, pp. 8203–8210.

[17] M. Fox and D. Long, ''PDDL2.1: An extension to PDDL for expressing temporal planning domains,'' *J. Artif. Intell. Res.*, vol. 20, pp. 61–124, Dec. 2003.

[18] R. P. Bonasso, R. J. Firby, E. Gat, D. Kortenkamp, D. P. Miller, and M. G. Slack, ''Experiences with an architecture for intelligent, reactive agents,'' *J. Experim. Theor. Artif. Intell.*, vol. 9, nos. 2–3, pp. 237–256, Apr. 1997.

[19] S. Matsuoka and T. Sawaragi, ''Recovery planning of industrial robots based on semantic information of failures and time-dependent utility,'' *Adv. Eng. Informat.*, vol. 51, Jan. 2022, Art. no. 101507.

[20] K. French, S. Wu, T. Pan, Z. Zhou, and O. C. Jenkins, ''Learning behavior trees from demonstration,'' in *Proc. Int. Conf. Robot. Autom. (ICRA)*, May 2019, pp. 7791–7797.

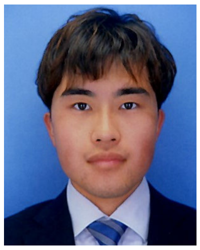
[21] M. Colledanchise, D. Almeida, and P. Ögren, ''Towards blended reactive planning and acting using behavior trees,'' in *Proc. Int. Conf. Robot. Autom. (ICRA)*, May 2019, pp. 8839–8845.

[22] M. Colledanchise and P. Ogren, *Behavior Trees in Robotics and AI: An Introduction*. Boca Raton, FL, USA: CRC Press, 2018.

[23] S. Yang, X. Mao, S. Wang, and Y. Bai, ''Extending behavior trees for representing and planning robot adjoint actions in partially observable environments,'' *J. Intell. Robotic Syst.*, vol. 102, no. 2, p. 36, Jun. 2021.

[24] Z. Han, D. Giger, J. Allspaw, M. S. Lee, H. Admoni, and H. A. Yanco, ''Building the foundation of robot explanation generation using behavior trees,'' *ACM Trans. Hum.-Robot Interact.*, vol. 10, no. 3, pp. 1–31, Sep. 2021.

[25] Z. Han, T. Williams, and H. A. Yanco, ''Mixed-reality robot behavior replay: A system implementation,'' in *Proc. AAAI Fall Symp. Artif. Intell. Hum.-Robot Interact. (AI-HRI)*, 2022.



KOKI NAKAMURA received the B.S. degree in mechatronics engineering from Meijo University, Aichi, Japan, in 2022. He is currently pursuing the M.S. degree in mechatronics engineering. His research interest includes human–robot interaction systems.

KOSUKE SEKIYAMA received the B.E., M.E., and Dr. (Eng.) degrees from Nagoya University, Japan, in 1992, 1994, and 1997, respectively. From 1998 to 2001, he was a Lecturer with the Science University of Tokyo. He was an Associate Professor with the University of Fukui, from 2001 to 2006. He joined Nagoya University as an Associate Professor, in 2006. Since 2019, he has been a Professor with the Department of Mechatronics Engineering, Meijo University. His research interests include collective robotics, multi-agent systems, swarm intelligence, human–robot interaction, and cognitive robotics. He is a member of Robotic Society of Japan (RSJ), Japan Society of Mechanical Engineering (JSME), and Society of Instrument and Control Engineers (SICE).