2026 IEEE International Conference on Robotics and Automation (ICRA 2026)
June 1-5, 2026. Vienna, Austria

Video-to-BT: Generating Reactive Behavior Trees from Human Demonstration Videos for Robotic Assembly

Xiwei Zhao[1,2*], Yiwei Wang[1*], Yansong Wu[1], Fan Wu[3†], Teng Sun[3], Zhonghua Miao[3], Sami Haddadin[4], Alois Knoll[1]

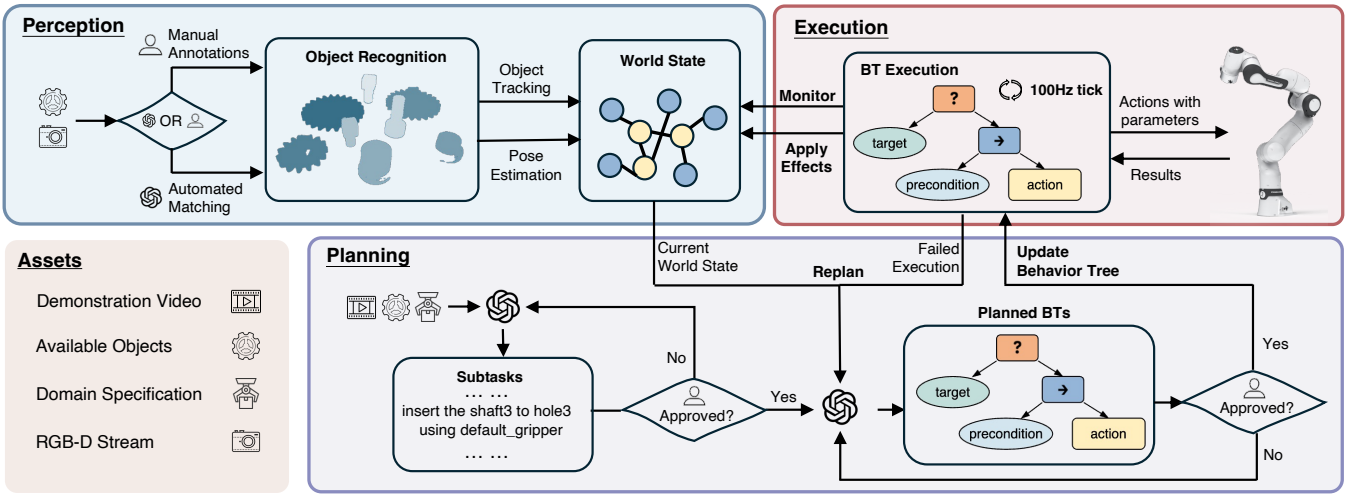

Fig. 1: **Overall framework**. The proposed framework operates through the synergy of three core modules: Planning, Perception, and Execution. Initially, the Planning Module generates BTs from video demonstrations, while the Perception Module maintains a real-time semantic understanding of the environment via the construction and update of the world state. The Execution Module then leverages the BTs and world state to command the robot's actions. This architecture forms a crucial closed loop: when the Execution Module detects a need for replanning based on perceptual data, it triggers the Planning Module to refine the BT, ensuring robust task completion.

***Abstract*— Modern manufacturing demands robotic assembly systems with enhanced flexibility and reliability. However, traditional approaches often rely on programming tailored to each product by experts for fixed settings, which are inherently inflexible to product changes and lack the robustness to handle variations. As Behavior Trees (BTs) are increasingly used in robotics for their modularity and reactivity, we propose a novel hierarchical framework, Video-to-BT, that seamlessly integrates high-level cognitive planning with low-level reactive control, with BTs serving both as the structured output of planning and as the governing structure for execution. Our approach leverages a Vision Language Model (VLM) to decompose human demonstration videos into subtasks, from which BTs are generated. During the execution, the planned BTs combined with real-time scene interpretation enable the system to operate reactively in the dynamic environment, while VLM-driven replanning is triggered upon execution failure. This closed-loop architecture ensures stability and adaptivity. We validate our framework on real-world assembly tasks through a series of experiments, demonstrating high planning reliability, robust performance in long-horizon assembly tasks, and strong generalization across diverse and perturbed conditions. Project website: https://video2bt.github.io/video2bt_page/**

I. Introduction

The pivotal shift from mass production to mass customization within modern, dynamic factory environments necessitates production lines that are not only reliable but also flexible [1]–[3]. Specifically, the principle of inherent safety demands high execution transparency and predictability, especially as human-robot collaboration becomes commonplace. Moreover, the goal of customization demands flexibility to manage high product diversity, while the operational environment requires robustness against shop-floor disturbances like component displacements or human interventions. Conventional robotic systems, fundamentally reliant on static and pre-programmed tasks, are thus ill-equipped for the procedural variations and dynamic uncertainties. Therefore, realizing the promise of smart manufacturing calls for a new generation of robotic systems, endowed with both the cognitive intelligence for high-level task planning and reasoning, as well as the adaptive mechanisms for robust and transparent low-level execution in dynamic environments.

Such robotic systems can be modeled as discrete-event dynamic systems (DEDS) [4] by abstracting the primitive action execution and state transitions as discrete events. Traditional DEDS supervisory controllers, such as Petri nets [5], [6] and finite state machines [7], [8], often face scalability limitations and provide limited support for recovery from disturbances [9]. Against this backdrop, the Behavior Trees (BTs), which represent policies in a hierarchical tree structure, have drawn increasing attention. Their modularity, human-readability, and reactivity make them well-suited to integrated planning and control in long-horizon tasks under dynamic environments. While there are existing methods

* Equal contribution.
† Corresponding author: Fan Wu (wufan@shu.edu.cn).
[1] Munich Institute of Robotics and Machine Intelligence (MIRMI), Technical University of Munich, Germany. [2] Aalto University, Espoo, Finland. [3] Shanghai University, Shanghai, China. [4] Mohamed Bin Zayed University of Artificial Intelligence, Abu Dhabi, UAE.

for automated BT generation, the practical use of BTs in long-horizon tasks still relies on deterministic settings and substantial user inputs. The BT generation from human-centric instructions and subsequent application of BTs to adaptive control remain largely underexplored.

The challenge of automatically constructing BTs from human-centric instructions finds a promising solution in the recent advancements of foundation models, such as Vision Language Models (VLMs) and Large Language Models (LLMs). The advanced faculties of the models for syntactic synthesis and logical inference [10] enable the direct generation of these abstract plans into formally specified, compositional structures such as BTs. Furthermore, these models possess unprecedented capabilities in common-sense reasoning and generalization, allowing them to interpret diverse, high-level instructions and decompose them into logical task sequences [11], [12]. While early works have explored using LLMs to generate BTs from simplified instructions [13], the full potential of leveraging VLMs to bridge rich, multi-modal perception directly with the generation of complex, structured BTs remains largely untapped.

To this end, we propose *Video-to-BT*, a framework that combines the generalization and reasoning capabilities of foundation models with the advantages of BTs to address long-horizon assembly tasks in dynamic environments. In our approach, a VLM serves as the brain for high-level planning: it infers the task sequence from a video demonstration and generates BTs as a structured representation of the sequential manipulation plan. These BTs are then employed as a modular and reactive control architecture, executing under perception-based scene interpretation that monitors the environment states. In this way, the robot receives instructions to initiate or suspend the primitive actions. To guarantee safe execution, the framework adopts a human-in-the-loop (HITL) workflow via a web-based interface, ensuring accessibility for non-experts. The presented framework is evaluated by a real world long-horizon assembly task. The results demonstrate the superiority of our framework, showcasing high planning accuracy, robust execution of assembly tasks, and strong reactivity to external disturbances in dynamic environments. To summarize, the main contributions of this work are as follows:

1) We propose a novel planning framework that translates human demonstration videos into structured, executable BTs for complex robotic assembly. The key to the planning framework is a three-stage pipeline that uniquely integrates VLM-driven task decomposition, parameterized plan formulation, and automated BT synthesis.
2) A supervisory control framework is developed that integrates a BT executor with real-time semantic perception to deliver precise, context-aware scheduling under dynamic conditions. By continuously monitoring the workspace and incorporating a recovery mechanism, the system possesses the capability to react to external disturbances.
3) The framework is validated in a systematic way. The results demonstrate that the framework achieves accurate planning across diverse demonstrations and sustains a high completion rate for long-horizon tasks, even under disturbances, highlighting the practical reliability and robustness of the proposed methods.

II. Related work

A. Adaptive Robotic Assembly in Dynamic Environments

The pursuit of enhanced flexibility for diverse tasks and robustness against environmental disturbances is a central challenge in modern smart manufacturing. While prominent approaches enhance adaptability in dynamic environments, such as Digital Twins (DTs) through high-fidelity simulation [14], [15] and Vision-Language-Action (VLA) models through powerful generalization [16]–[18], they both demand substantial upfront investment in terms of expert labor and resources. DTs demand intensive, expert-driven labor for creating simulators and digital assets [19], [20], whereas VLAs face a critical bottleneck in their dependency on large-scale datasets [21], [22]. These challenges are especially acute for complex assembly, which demands contact-rich tactile data that is notoriously scarce and difficult to acquire. To circumvent the need for extensive task-specific training or data collection, a hierarchical framework is proposed that leverages open-vocabulary foundation models to dynamically sequence a library of robust skill primitives [23].

B. VLM-based Task Planning in Robotics

VLMs have shown significant promise for robotic task planning, due to their powerful environmental perception, common sense knowledge, and strong generalization, which enable the generation of multi-step online task plans, as seen in embodied reasoning [24]–[27]. However, the generalization capability derived from large-scale, general-purpose datasets does not readily transfer to precision-critical tasks such as assembly. Instead of attempting to generate plans directly, a common strategy is to leverage the multi-modal understanding of VLMs to interpret human-provided instructions, such as diagrams or manuals, to derive high-level task sequences [28], [29]. Although this approach is effective, its primary limitation is the dependency on formal documentation, which precludes its use for novel or undocumented tasks. To bridge this gap, this work introduces a novel planning module that leverages a VLM to interpret human assembly demonstration videos as a rich, multi-modal source of instruction. From these demonstrations, our approach extracts nuanced procedural knowledge, offering a more robust and generalizable solution that enables learning from non-expert demonstrators and bypasses the need for formal documentation.

C. Behavior Tree in Robotics

Compared to other control architectures such as Petri nets [5], [6] and finite state machines [7], [8], BTs offer a comparable hierarchical organization and modularity, while providing the additional advantage of reactivity [9]. Consequently, BTs have been increasingly applied in task planning and task execution. On the planning side, existing studies have explored both analytical formulations [30], [31] and learning-based approaches [32], [33] for BT generation, yet these methods still rely heavily on pre-defined planning requirements and extensive manual programming. While there are some studies that address it by leveraging foundation models to automatically generate BTs [13], [34], [35],

they still require substantial user input to specify planning requirements. On the control side, some works [36] [37] have applied BTs to accomplish long-horizon tasks under fully deterministic settings, lacking adaptivity against uncertainty. In contrast, some studies [38]–[40] have emphasized the adaptive execution, but reliance on handcrafted design restricts their applicability. Inspired by these works, our framework integrates the automated BT generation based on task decomposition, supervisory control for long-horizon execution, and reactive control for disturbance handling to enable robust task execution in a dynamic environment.

III. Methods

To address the aforementioned problems, we proposed a framework that consists of **Planning**, **Perception**, and **Execution**, as illustrated in Fig. 1. This tight coupling between high-level planning, real-time scene interpretation, and low-level control constructs a closed-loop system that enables scalable and interpretable solutions for long-horizon assembly tasks.

A. Preliminaries

System Setup: The system setup consists of providing resources, specifying the domain, and initializing the world state, which are defined as follows:

- **Resources Provision** (V, O): V is the human demonstration video. O denotes the objects available, including both manipulable components and usable tools.
- **Domain Specification** $\mathcal{D} = (\mathcal{P}, \mathcal{C}, \mathcal{R}, \mathcal{A}, \mathcal{S})$: $\mathcal{P}$ is the set of unary predicates describing the object properties. Both $\mathcal{C}$ and $\mathcal{R}$ are sets of binary predicates, characterizing the inter-object constraints and relations, respectively. $\mathcal{A}$ is the set of available action primitives. $\mathcal{S}$ denotes the set of skill symbols, each of which represents a capability to accomplish a specific task. All elements in the domain vocabulary can be instantiated by being assigned to concrete objects. A full set of concrete examples is provided in Table I.
- **World State Initialization** ω: The world state is modeled as $\omega = (P, R)$, where P, R comprise object properties and inter-object relations, respectively. In essence, ω captures the facts that hold in the environment. ω is initialized by loading a predefined specification of object properties and relations that characterize the task setup.

Behavior Trees: A BT is a directed rooted tree whose internal nodes serve as *control flow nodes* and leaf nodes serve as *execution nodes*, providing a hierarchical structure for task specification [9]. There are four types of nodes within this framework:

- Sequence: A control flow node graphically represented as "$\rightarrow$" that returns `Success` only if all children succeed; otherwise, it returns the status of the first child that returns `Failure` or `Running`.
- Selector (Fallback): A control flow node graphically represented as "?" that returns the status of the first child that is `Success` or `Running`. It returns `Failure` only if all children fail.
- Condition: An execution node that evaluates a proposition, i.e., a constraint, a property, or a relation. It deterministically returns either `Success` or `Failure`.

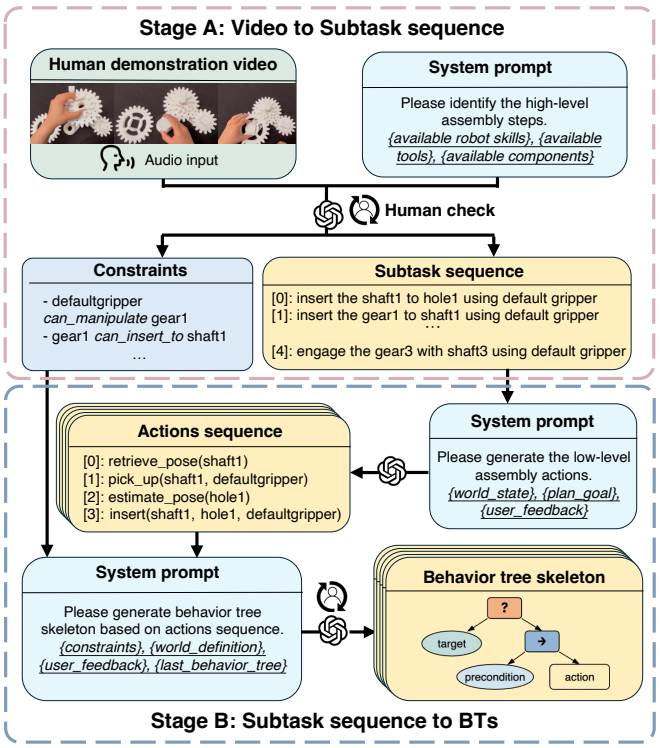

Fig. 2: **The planning module** involves two VLM-driven stages. Stage A generates subtasks and inter-object constraints by interpreting a human demonstration video; stage B structures and compiles the subtasks to generate the final BT. The human verification is also embedded in both generation stages to ensure the safety and reliability.

- Action: An execution node that invokes an action defined as $a_i(O_i)$, where $a_i \in \mathcal{A}$ and $O_i \subseteq O$. It returns `Running` while being executed asynchronously, `Failure` on error, and `Success` on completion. Upon success, its effects are updated to the world state.

A BT executes by recursively propagating ticks in a depth-first, left-to-right order. Tick propagation is typically repeated at a fixed control frequency.

When a condition node is placed under a selector, its success causes the selector to skip the remaining children, implying the target of the subsequent execution. When a condition node or a subtree is placed under a sequence, its success is required for the execution to proceed, making it function as a precondition. This framework follows the principle that every action node should be embedded in an action unit (e.g., the BT shown in Fig. 2) that contains a target and optional precondition(s).

B. Planning: BT generation from video demonstration

The planning module leverages the generalization and reasoning capabilities of pre-trained VLMs to translate human-oriented instructional videos into robot-oriented BTs. Unlike the traditional role of an expert programmer, our module allows the operator to serve as a robot mentor, who simply demonstrates tasks in a human-centered manner, reviews the learned results, and provides minimal HITL guidance when necessary. As illustrated in Fig. 2, the planning module operates in two stages: video interpretation and BT generation.

Video interpretation: Human demonstration videos are

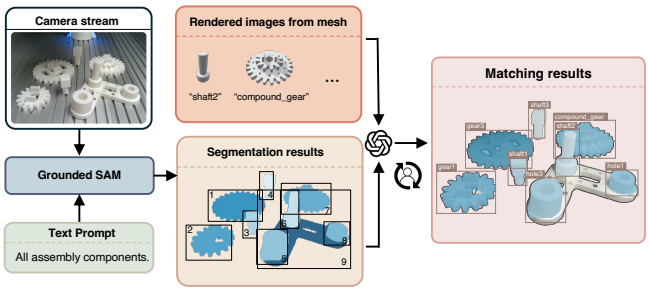

Fig. 3: **Objects recognition** includes two steps. First, the Grounded SAM [41] model is employed for coarse segmentation and numerical annotation of assembly components. Subsequently, a VLM matches these annotated masks with mesh-rendered images to obtain the final matched masks and their corresponding names.

recorded from an action-centric view to capture a complete assembly procedure. Each video is accompanied by an auxiliary audio description, which is intentionally multilingual and stylistically diverse. Illustrative video samples are provided in Fig. 6. To process the videos, the key frames and the audio are first extracted from the video. The key frames are selected manually to ensure that they capture the essential steps of the assembly. The extracted audio is then converted to text, providing a complementary prompt for further reasoning. Based on key frames and audio text, a VLM agent can infer the sequence of subtasks $\langle \tau_1, \ldots, \tau_N \rangle$ and constraints C among relevant objects, grounded in the set of objects O and available skills $\mathcal{S}$.

$$\tau_i = (s_i, o_{i1}, o_{i2}),\ s_i \in \mathcal{S}, \tag{1}$$

$$C = \{c(o_1, o_2) \mid c \in \mathcal{C},\ o_1, o_2 \in O\}, \tag{2}$$

where τ_i defines one subtask in the integrated assembly procedure and C encodes intrinsic constraints among objects (e.g., o_1 *can_insert_to* o_2), forming a crucial prerequisite for the BT generation.

BT generation: The planning module generates BTs through a sequential process, where each subtree b_i is created and verified individually. For each given subtask τ_i, the process begins with an LLM agent planning an action sequence $\mathbf{a}_i = \langle a_{i1}(O_{i1}), \ldots, a_{iM}(O_{iM}) \rangle$. Subsequently, as $\mathbf{a}_i$ serves as a blueprint, b_i is generated by an LLM agent, with reference to the domain $\mathcal{D}$, the constraints C, and the world state ω_i that describes the environment at the beginning of each subtask. The generated BT is then verified in a HITL process. Upon the generation of a correct subtree b_i, a BT simulator is utilized to perform a virtual tick, thereby transitioning the world state from ω_i to ω_{i+1}. This resultant state ω_{i+1} serves as a prerequisite context for the synthesis of the subsequent subtree b_{i+1}. This iterative procedure is applied across the entire sequence of subtasks, culminating in the final BT sequence $\Pi = \langle b_1, \ldots, b_N \rangle$.

C. Perception: Semantic and dynamic scene understanding

The perception module captures environmental changes caused by external disturbances through object recognition and continuous scene interpretation, allowing the world state ω^t to be updated accordingly and to remain a coherent representation of the dynamic environment.

Object recognition can be accomplished either through a VLM-based matching pipeline or by manual annotation. As illustrated in Fig. 3, the VLM-based pipeline leverages Grounded SAM [41] as an object detector and the generalization capabilities of VLMs. Guided by a simple text prompt, the detector extracts object masks and annotates them with a number. These annotated masks, together with images rendered from the component meshes, are then provided to a VLM agent to match the scene objects with those in object set O, followed by a human check to ensure correctness. Alternatively, a web interface for manual annotation is developed that allows users to interactively mark each object in the scene corresponding to O.

Runtime scene interpretation is achieved through two visual modules: (i) The first module $\mathcal{M}_1$ utilizes SAM2 [42] to segment the scene based on the result of object recognition and maintain object tracking throughout the assembly process. In doing so, it produces masks of the objects, which provide an approximate representation of the object positions in the scene. (ii) The second module $\mathcal{M}_2$ utilizes FoundationPose [43] to maintain 6D-pose estimations of the objects involved in the current action $a_i(O_i)$, providing pose information for the executor.

World state update is performed by an UPDATESTATE operation, which applies the following rules based on $\mathcal{M}_1$ and $\mathcal{M}_2$ to update ω^t in runtime.

- **Object position invariance.** For objects not involved in the current action $a_i(O_i)$, any discrepancy in position identified by $\mathcal{M}_1$ will trigger an update of the recorded mask and position.
- **Relation validity.** For each relation $r(o_i, o_j) \in R$, if a relative displacement of o_i and o_j is detected by $\mathcal{M}_1$, the relation is deemed invalid and removed from R, i.e., $R^t \leftarrow R^{t-1} \setminus \{r(o_i, o_j)\}$.
- **Pose consistency.** For objects involved in the current action $a_i(O_i)$, if $\mathcal{M}_2$ detects that an object $o_j \in O_i$ is moved unexpectedly, the pose of o_j should be considered unknown for the executor before the newly detected pose is retrieved. Accordingly, the object property is updated as $P^t \leftarrow P^{t-1} \setminus \{\textit{pose\_is\_known}(o_j)\}$.

By applying these rules, the world state is continuously maintained as a coherent and reliable representation of the environment.

D. Execution: Reactive BT with recovery mechanism

The execution module carries out the planned BTs based on the real-time scene interpretation, enabling the assembly task to be accomplished in a dynamic environment. As illustrated in Algorithm 1, each b_i in the BT sequence Π is sequentially extended and executed, with continuous world state monitoring and the recovery mechanism to ensure robustness.

BT extension: Each subtree b_i corresponds to the subtask τ_i, whose successful accomplishment yields an inter-object relation r_i that represents the achieved subgoal (for instance, the accomplishment of *insert(gripper, shaft, gear)* establishes the relation *is_inserted_to(gear, shaft)*). During the execution of b_i, the validity of the relations established by previously executed subtrees must be preserved. To ensure this, these relations $(r_1, \ldots, r_{i-1})$ are incorporated as preconditions of b_i by placing them as the leading children of a sequence node, resulting in an extended BT $\hat{b}_i$ defined as

SEQUENCE$(r_1, \ldots, r_{i-1}, b_i)$. An example of the extended BT is illustrated in Fig. 4.

World state monitoring: Throughout the process, the world state ω^t is continuously updated by the operation UPDATESTATE, while the BTs are executed sequentially in synchrony. The extended BT $\hat{b_i}$ is ticked at a frequency f until success. At every tick, the condition nodes evaluate propositions against ω^t, thereby verifying that both the previously achieved subgoals and the preconditions of the current action remain preserved.

Recovery mechanism: When a precondition fails, the ongoing action will be suspended and the recovery mechanism will be triggered. The tick propagation will search for an action unit whose target effect can restore the violated condition. If such a unit exists, its action (e.g., retrieving pose) will be enforced before resuming the prior execution. Otherwise, self-recovery is impossible and the BT will return `Failure`; the mechanism thereby will identify the failed node and triggers replanning based on ω^t: if a previously achieved subgoal r_j with $j < i$ has been violated, the process will roll back to the stage j and replan b_j; otherwise, replan the current subtree b_i.

Integrated with guarded preconditions, real-time evaluation, and corrective recovery, the system remains reactive to the dynamic environment, ensuring a robust execution for long-horizon tasks under disturbances.

Algorithm 1: Reactive Execution of BTs

Input: Subtree sequence $\Pi = \langle b_1, \ldots, b_N \rangle$
Data: Realtime World State ω^t

Thread A (Reactive BT Execution Loop):
for $i \leftarrow 1$ **to** N **do**
  repeat
    $\hat{b_i} \leftarrow$ SEQUENCE$(r_1, \ldots, r_{i-1}, b_i)$;
    `status` $\leftarrow$ `Running`;
    while `status` = `Running` **do**
      `status` $\leftarrow$ TICK$(\hat{b_i}, \omega^t, f)$; // self-recovery embedded
    end
    if `status` = `Failure` **then**
      $n_{\text{fail}} \leftarrow$ FAILEDNODE$(\hat{b_i})$;
      if $n_{\text{fail}} = r_j$ *with* $j < i$ **then**
        $b_j \leftarrow$ REPLAN(b_j, ω^t);
        $i \leftarrow j$;
        break; // rollback to stage j
      else if $n_{\text{fail}} = b_i$ **then**
        $b_i \leftarrow$ REPLAN(b_i, ω^t); // replan the current BT
      end
  until `status` = `Success`;
end

Thread B (world-state maintenance):
repeat
  $a_i(O_i) \leftarrow$ CURRENTACTION();
  $\omega^t \leftarrow$ UPDATESTATE$(\omega^{t-1}, a_i(O_i), \mathcal{M}_1, \mathcal{M}_2)$;
until THREAD A ENDS;

IV. EXPERIMENTS AND RESULTS

The proposed framework is specifically designed to address the challenges of complex, long-horizon tasks that involve multiple constraints and dynamic environmental uncertainties. Accordingly, the subsequent experiments are designed to address the following research questions:

RQ1: To what extent is our proposed video-to-BT planning module generalizable and reliable?

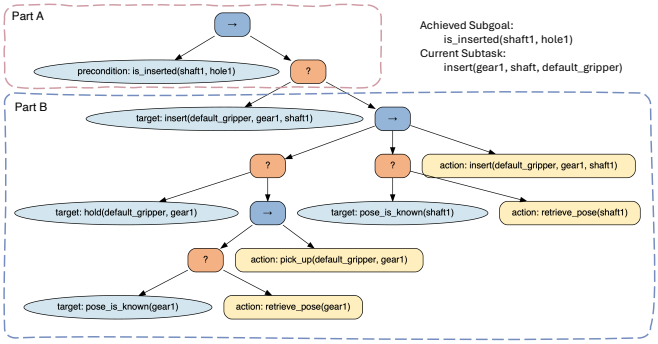

Fig. 4: **Example of an extended executable BT** that highlights its two constituent parts, i.e., Part A: the extension mechanism; Part B: BT generated by a VLM for a subtask.

RQ2: How effectively can our perception module perform automated object recognition?

RQ3: How robust is our reactive BT in execution under external disturbances?

A. Experiment Setup

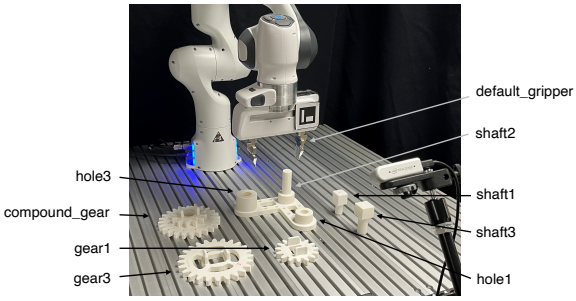

Fig. 5: **Experiment setup.**

As illustrated in Fig. 5, the experiment setup consists of a Franka Emika Panda robot, an Intel Realsense D435i camera, and a gearset from the Siemens Robot Assembly Challenge. The fundamental action set employed by the BTs is derived from the taxonomy of skills described in [23]. The domain is specified in the Table I.

TABLE I: Domain specification

Category	Signature
$\mathcal{P}$	is_empty, pose_is_known
$\mathcal{C}$	can_manipulate, can_insert_to, can_engage_with, can_place_on
$\mathcal{R}$	is_inserted_to, is_engaged_with, is_placed_on, hold
$\mathcal{A}$	change_tool, pick_up, put_down, insert, engage, place, retrieve_pose
$\mathcal{S}$	insert, engage, place

B. BT generation from demonstration (*RQ1*)

Experiment Design: To facilitate a comprehensive evaluation of our planning module, 25 human demonstration videos have been prepared. The accompanying instructions are intentionally diverse, spanning multiple languages and varied narrative styles (samples in Fig. 6). The evaluation assesses the module from three key aspects: its generalization to diverse video inputs, the reliability endowed by the HITL process, and its compatibility with varying foundation models. For each model tested, our analysis distinguishes between the performance at two critical stages: the initial response from direct prompting and the refined response following a single round of human feedback.

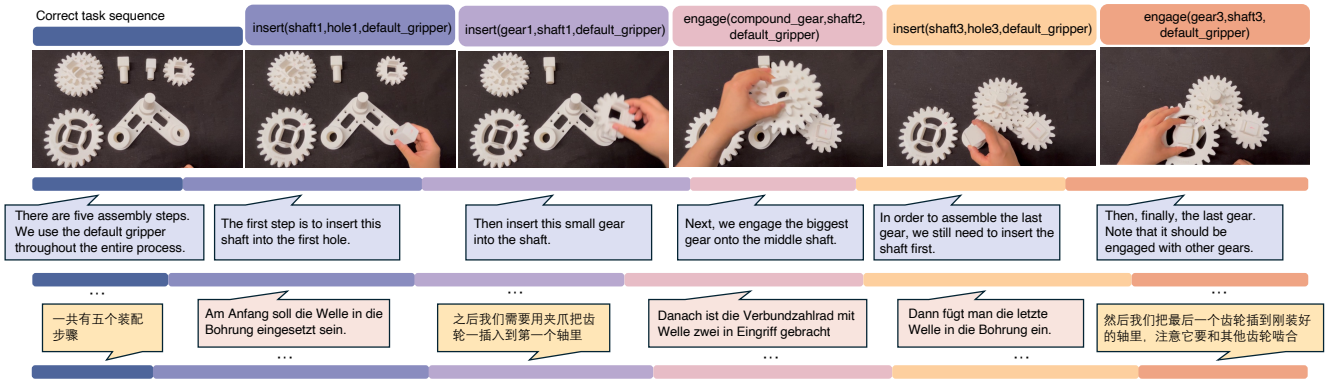

Fig. 6: **Human demonstration video samples.** The videos showcase assembly procedures, supplemented by flexible audio descriptions in multiple languages (e.g., English, German, and Chinese) and varied narrative styles.

The quality of generated BTs from a single video is then quantified using the following three metrics, designed to assess both the semantic correctness of its decomposed subtasks as well as the logical and syntactic integrity of the generated BTs:

- Task Decomposition Accuracy (TDA). The percentage of generated subtasks that correctly correspond to the video. A subtask is deemed correct if its semantic elements (skill, components, and tool) are consistent with the video demonstration.
- Logical Coherence Rate (LCR): The percentage of generated BTs that exhibit a logically coherent sequence of actions. A BT is considered coherent if it adheres to procedural constraints and preconditions, such as requiring an object to be grasped before it can be placed.
- Syntactic Validity Rate (SVR): The percentage of generated BTs that are syntactically valid. This metric verifies that the BT structure adheres to its formal grammar and formatting rules.

Results: The evaluation results are summarized in the Table II, presenting the average rate for each metric calculated across all 25 test videos.

TABLE II: BT generation result using different VLMs

	GPT-4o			Gemini-1.5-flash			Qwen-2.5VL-72b		
Method	**TDA**	**LCR**	**SVR**	**TDA**	**LCR**	**SVR**	**TDA**	**LCR**	**SVR**
Initial response	98%	99%	100%	81%	61%	99%	76%	80%	100%
Refined response	100%	100%	100%	94%	88%	100%	79%	100%	100%

The results confirm the ability of the VLM-based planning module to generate high-quality BTs by combining high-performance autonomous generation with a robust HITL safeguard. In task decomposition, initial TDA scores were impacted by visual ambiguities such as poor component visibility or ambiguous verbal instructions. However, a single refinement proved highly effective, raising the accuracy for GPT-4o to 100% and Gemini-1.5-flash to 94%. This process successfully corrects initial imperfections to ensure an accurate final task sequence.

For the subsequent BT structural generation provided with the correct task sequence, GPT-4o achieved an excellent initial LCR of 99%. The HITL process also bridged performance gaps for other smaller models, elevating the LCRs of Gemini-1.5-flash from 61% to 88% and Qwen-2.5VL-72b from 80% to 100%. This demonstrates the module's capacity to consistently deliver logically coherent and executable BTs.

C. Semantic perception (*RQ2*)

The perception module allows both manual annotation and a VLM-based matching pipeline as illustrated in Fig. 3 for object recognition. Manual annotation consistently provides correct and stable recognition. As object recognition constitutes the foundation of subsequent scene perception, the experiment aims to examine how reliably the VLM-based matching pipeline can substitute for the manual annotation, with GPT-4o employed as the VLM agent.

Experiment Design: This experiment evaluates how two key factors influence the success rate of the mask-to-name matching process. The first factor is the number of interaction rounds that we assign to 0, 1, 3, and 5 to represent varying levels of human assistance. The second is the composition of the human feedback, where we vary the number of correct mask-name pairs (e.g., one versus multiple) provided in each round to represent different information densities. For each experimental configuration, we measure the overall success rate, defined as the proportion of 15 complete trials that result in a perfectly correct match for all assembly components. The aggregated results are presented in Table III.

TABLE III: Accuracy rate of GPT-4o matching

	Number of Refinement Steps			
Information Density	**0**	**1**	**3**	**5**
one pair	4/15	6/15	12/15	15/15
multiple pairs	4/15	13/15	15/15	15/15

Results: Table III indicates that iterative design of the pipeline is robust, consistently reaching a 15/15 correct match within five cycles. It is also observed that more informative feedback accelerates this process, with richer prompt details leading to faster convergence. Overall, these findings confirm that the automated pipeline performs on par with manual annotation and serves as a stable alternative.

D. Real-world BT Execution (*RQ3*)

Provided with the correctly planned BTs and segmentation results from the previous stages, the execution of the BTs is then deployed on the Franka robot to validate the robustness and reactivity of our framework in executing assembly tasks in the dynamic environment.

Experiment Design: As illustrated in Fig. 7, the experiments are designed with three difficulty levels, characterized

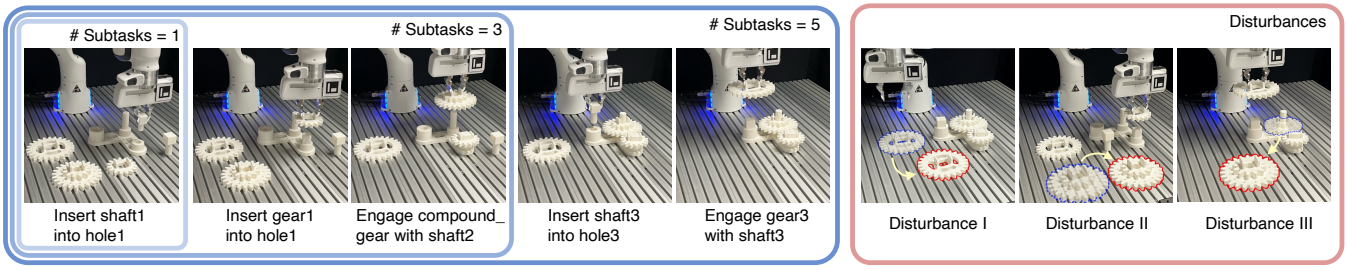

Fig. 7: **Demonstration of the assembly procedure and disturbances**. **Left**: Assembly execution procedures consisting of one, three, and five subtasks. With the growth of task length, the number of involved objects also increases, making the overall execution more challenging. **Right**: Examples of external disturbances: *Disturbance I*, moving the gear3 while the robot is approaching it; *Disturbance II*, moving the compound_gear when the robot tries to insert the gear1 into the shaft1; *Disturbance III*, extracting the compound_gear from the shaft2 before the robot attempts to insert the gear3 into the shaft3.

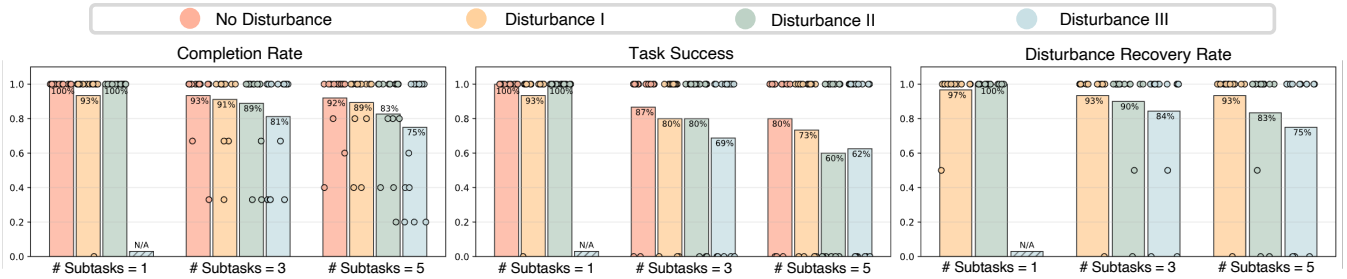

Fig. 8: **Execution results**. Performance comparison across varied task lengths and disturbance types. The bar heights indicate the average value, with individual trial results shown as scatter points.

by assembly tasks consisting of one, three, and five subtasks. In addition, four cases are considered to verify the disturbance resilience of the system: *No external disturbance*; *Disturbance I*, displacement of the component involved in the current action; *Disturbance II*, displacement of components that have not yet been involved in any actions; and *Disturbance III*, violation of a previously achieved goal. Each experimental setting was run for 15 trials.

The following quantitative evaluation metrics are measured for each single trial:

- Task Success (TS): Whether the entire assembly task is completed successfully, requiring full execution of all actions and proper handling of perturbations.
- Completion Rate (CR): The percentage of accomplished subtasks.
- Disturbance Recovery Rate (DRR): The percentage of disturbances that are successfully addressed.

Results: The results are shown in Fig. 8. Without disturbance, both average CR and average TS among the 15 trials remain high, above 90% and 80% respectively, confirming that the system can reliably execute long-horizon tasks. Under disturbances, the performance degrades moderately: the average CR exceeds 75% and the average TS exceeds 60% for all cases, highlighting the robust performance and effective recovery under diverse disturbances.

To further understand the disturbance robustness of the system, an analysis of failed recovery cases was conducted:

- *Disturbance I* is handled based on precise pose tracking of a single object at the primitive action level. The main reason for the failed handling stems from occasional estimation deviations, which cause execution errors in individual actions.
- *Disturbance II* and *Disturbance III* are handled based on the object tracking throughout the entire process. Nevertheless, the average DRR exhibits a noticeable decline from 100% to 83% and from 84% to 75%, respectively, as the number of subtasks and involved objects increases, reflecting the rising likelihood of tracking losses or missassignments. Such errors cause misalignments between the scene graph and the physical environment, leading to incorrect condition evaluations and ultimately execution failure.

The failure case analysis points to the limitation of our framework: perception-related errors occur more frequently as the task length increases, which causes a decrease in success rate for longer horizons; but it also suggests that advances in perception modules will enhance the robustness and scalability of the entire system.

V. Conclusions

This paper presents *Video-to-BT*, a holistic framework enabling non-experts to deploy robots for complex, long-horizon assembly tasks. Our approach effectively bridges the gap between high-level planning and real-world execution by leveraging foundation models to translate demonstration videos into executable BTs. Our framework consists of three tightly-coupled modules. Initially, a robust planning module generates an executable BTs. Then, the generated BTs are executed under the run-time guidance of the semantic perception module that detects real-time disturbances.The tight synergy between planning and perception enables reactive and robust task execution in dynamic environments. In future work, we will focus on extending the framework's applicability to a broader range of manipulation tasks.

References

[1] M. Pantano, T. Eiband, and D. Lee, "Capability-based frameworks for industrial robot skills: a survey," in *2022 IEEE 18th International*

Conference on Automation Science and Engineering (CASE). IEEE, 2022, pp. 2355–2362.

[2] J. Xu, Q. Sun, Q.-L. Han, and Y. Tang, "When embodied ai meets industry 5.0: Human-centered smart manufacturing," *IEEE/CAA Journal of Automatica Sinica*, vol. 12, no. 3, pp. 485–501, 2025.

[3] K. Nikiforidis, A. Kyrtsoglou, T. Vafeiadis, T. Kotsiopoulos, A. Nizamis, D. Ioannidis, K. Votis, D. Tzovaras, and P. Sarigiannidis, "Enhancing transparency and trust in ai-powered manufacturing: A survey of explainable ai (xai) applications in smart manufacturing in the era of industry 4.0/5.0," *ICT Express*, vol. 11, no. 1, pp. 135–148, 2025.

[4] C. G. Cassandras and S. Lafortune, *Introduction to discrete event systems*. Springer, 2007.

[5] R. Zurawski and M. Zhou, "Petri nets and industrial applications: A tutorial," *IEEE Transactions on industrial electronics*, vol. 41, no. 6, pp. 567–583, 1994.

[6] H. Costelha and P. Lima, "Modelling, analysis and execution of robotic tasks using petri nets," in *2007 IEEE/RSJ international conference on intelligent robots and systems*. IEEE, 2007, pp. 1449–1454.

[7] M. Foukarakis, A. Leonidis, M. Antona, and C. Stephanidis, "Combining finite state machine and decision-making tools for adaptable robot behavior," in *International conference on universal access in human-computer interaction*. Springer, 2014, pp. 625–635.

[8] D. C. Conner and J. Willis, "Flexible navigation: Finite state machine-based integrated navigation and control for ros enabled robots," in *SoutheastCon 2017*. IEEE, 2017, pp. 1–8.

[9] M. Colledanchise and P. Ögren, *Behavior trees in robotics and AI: An introduction*. CRC Press, 2018.

[10] N. Nascimento, E. Guimaraes, S. S. Chintakunta, and S. A. Boominathan, "Llm4ds: Evaluating large language models for data science code generation," *arXiv preprint arXiv:2411.11908*, 2024.

[11] W. Huang, F. Xia, T. Xiao, H. Chan, J. Liang, P. Florence, A. Zeng, J. Tompson, I. Mordatch, Y. Chebotar *et al.*, "Inner monologue: Embodied reasoning through planning with language models," in *Conference on Robot Learning*. PMLR, 2023, pp. 1769–1782.

[12] K. Black, M. Nakamoto, P. Atreya, H. Walke, C. Finn, A. Kumar, and S. Levine, "Zero-shot robotic manipulation with pre-trained image-editing diffusion models," in *NeurIPS 2023 Workshop on Goal-Conditioned Reinforcement Learning*.

[13] J. Ao, F. Wu, Y. Wu, A. Swiki, and S. Haddadin, "Llm-as-bt-planner: Leveraging llms for behavior tree generation in robot task planning," in *2025 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2025, pp. 1233–1239.

[14] C. Zhang, G. Zhou, D. Ma, Z. Wang, and Y. Zou, "Digital twin-driven multi-dimensional assembly error modeling and control for complex assembly process in industry 4.0," *Advanced Engineering Informatics*, vol. 60, p. 102390, 2024.

[15] J. Shi, X. Liu, M. Su, Y. Yi, H. Yang, and W. Cai, "A digital twin-based smart assembly for cable-driven parallel robots," *Advanced Engineering Informatics*, vol. 68, p. 103623, 2025.

[16] O. Mees, D. Ghosh, K. Pertsch, K. Black, H. R. Walke, S. Dasari, J. Hejna, T. Kreiman, C. Xu, J. Luo *et al.*, "Octo: An open-source generalist robot policy," in *First Workshop on Vision-Language Models for Navigation and Manipulation at ICRA 2024*, 2024.

[17] M. J. Kim, K. Pertsch, S. Karamcheti, T. Xiao, A. Balakrishna, S. Nair, R. Rafailov, E. P. Foster, P. R. Sanketi, Q. Vuong *et al.*, "Openvla: An open-source vision-language-action model," in *Conference on Robot Learning*. PMLR, 2025, pp. 2679–2713.

[18] K. Black, N. Brown, D. Driess, A. Esmail, M. Equi, C. Finn, N. Fusai, L. Groom, K. Hausman, B. Ichter *et al.*, "π0: A vision-language-action flow model for general robot control. corr, abs/2410.24164, 2024. doi: 10.48550," *arXiv preprint ARXIV.2410.24164*.

[19] J. Reif, D. Dittler, M. S. Gill, T. Farkas, V. Stegmaier, F. Gehlhoff, T. Kleinert, and M. Weyrich, "An expert survey on models and digital twins," *arXiv preprint arXiv:2506.17313*, 2025.

[20] I. A. Arin, H. L. H. S. Warnars, D. F. Murad *et al.*, "A systematic literature review of recent trends and challenges in digital twin implementation," in *2023 10th International Conference on ICT for Smart Society (ICISS)*. IEEE, 2023, pp. 1–10.

[21] K. Kawaharazuka, J. Oh, J. Yamada, I. Posner, and Y. Zhu, "Vision-language-action models for robotics: A review towards real-world applications," *TechRxiv, August*, vol. 12, 2025.

[22] Y. Wu, X. Chen, Y. Chen, H. Sadeghian, F. Wu, Z. Bing, S. Haddadin, A. König, and A. Knoll, "Sharedassembly: A data collection approach via shared tele-assembly," *arXiv preprint arXiv:2503.12287*, 2025.

[23] L. Johannsmeier, S. Schneider, Y. Li, E. Burdet, and S. Haddadin, "A process-centric manipulation taxonomy for the organization, classification and synthesis of tactile robot skills," *Nature Machine Intelligence*, vol. 7, no. 6, pp. 916–927, 2025.

[24] Y. Ji, H. Tan, J. Shi, X. Hao, Y. Zhang, H. Zhang, P. Wang, M. Zhao, Y. Mu, P. An, X. Xue, Q. Su, H. Lyu, X. Zheng, J. Liu, Z. Wang, and S. Zhang, "Robobrain: A unified brain model for robotic manipulation from abstract to concrete," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2025, pp. 1724–1734.

[25] R. Yang, H. Chen, J. Zhang, M. Zhao, C. Qian, K. Wang, Q. Wang, T. V. Koripella, M. Movahedi, M. Li *et al.*, "Embodiedbench: Comprehensive benchmarking multi-modal large language models for vision-driven embodied agents," in *Forty-second International Conference on Machine Learning*.

[26] W. Zhang, M. Wang, G. Liu, X. Huixin, Y. Jiang, Y. Shen, G. Hou, Z. Zheng, H. Zhang, X. Li *et al.*, "Embodied-reasoner: Synergizing visual search, reasoning, and action for embodied interactive tasks," *arXiv preprint arXiv:2503.21696*, 2025.

[27] H. Fang, M. Zhang, H. Dong, W. Li, Z. Wang, Q. Zhang, X. Tian, Y. Hu, and H. Li, "Robix: A unified model for robot interaction, reasoning and planning," *arXiv preprint arXiv:2509.01106*, 2025.

[28] C. Tie, S. Sun, J. Zhu, Y. Liu, J. Guo, Y. Hu, H. Chen, J. Chen, R. Wu, and L. Shao, "Manual2skill: Learning to read manuals and acquire robotic skills for furniture assembly using vision-language models," in *Proceedings of Robotics: Science and Systems (RSS)*, 2025.

[29] S. Li, Z. Yan, Z. Wang, and Y. Gao, "Vlm-msgraph: Vision language model-enabled multi-hierarchical scene graph for robotic assembly," *Robotics and Computer-Integrated Manufacturing*, vol. 94, p. 102978, 2025.

[30] M. Colledanchise, D. Almeida, and P. Ögren, "Towards blended reactive planning and acting using behavior trees," in *2019 international conference on robotics and automation (ICRA)*. IEEE, 2019, pp. 8839–8845.

[31] E. Scheide, G. Best, and G. A. Hollinger, "Synthesizing compact behavior trees for probabilistic robotics domains," *Autonomous Robots*, vol. 49, no. 1, p. 3, 2025.

[32] J. Styrud, M. Iovino, M. Norrlöf, M. Björkman, and C. Smith, "Combining planning and learning of behavior trees for robotic assembly," in *2022 International Conference on Robotics and Automation (ICRA)*. IEEE, 2022, pp. 11 511–11 517.

[33] C. Deng, C. Zhao, Z. Liu, J. Zhang, Y. Wu, Y. Wang, H. Cheng, and X. Yi, "Learning behavior trees by evolution-inspired approaches," in *Proceedings of the Companion Conference on Genetic and Evolutionary Computation*, 2023, pp. 275–278.

[34] S. Merino-Fidalgo, C. Sánchez-Girón, E. Zalama, J. Gómez-García-Bermejo, and J. Duque-Domingo, "Behavior tree generation and adaptation for a social robot control with llms," *Robotics and Autonomous Systems*, p. 105165, 2025.

[35] Q. Meng, Y. Wang, C. Feng, P. Li, and X. Xia, "Behavior generation for heterogeneous agents in urban simulation deduction: A multi-stage approach based on large language models," in *2025 IEEE 26th China Conference on System Simulation Technology and its Applications (CCSSTA)*. IEEE, 2025, pp. 456–462.

[36] S. C. Akkaladevi, M. Propst, K. Deshpande, M. Hofmann, and A. Pichler, "Towards a behavior tree based robotic skill execution framework for human robot collaboration in industrial assembly," in *2024 10th International Conference on Automation, Robotics and Applications (ICARA)*. IEEE, 2024, pp. 18–22.

[37] N. Wake, A. Kanehira, J. Takamatsu, K. Sasabuchi, and K. Ikeuchi, "Vlm-driven behavior tree for context-aware task planning," *arXiv preprint arXiv:2501.03968*, 2025.

[38] Y. Wu, F. Wu, L. Chen, K. Chen, S. Schneider, L. Johannsmeier, Z. Bing, F. Abu-Dakka, A. Knoll, and S. Haddadin, "1 khz behavior tree for self-adaptable tactile insertion," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2024.

[39] J. Cloete, W. Merkt, and I. Havoutis, "Adaptive manipulation using behavior trees," *arXiv preprint arXiv:2406.14634*, 2024.

[40] Y. Wu, Z. Chen, F. Wu, L. Chen, L. Zhang, Z. Bing, A. Swikir, S. Haddadin, and A. Knoll, "Tacdiffusion: Force-domain diffusion policy for precise tactile manipulation," in *2025 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2025, pp. 11 831–11 837.

[41] T. Ren, S. Liu, A. Zeng, J. Lin, K. Li, H. Cao, J. Chen, X. Huang, Y. Chen, F. Yan *et al.*, "Grounded sam: Assembling open-world models for diverse visual tasks," *arXiv preprint arXiv:2401.14159*, 2024.

[42] N. Ravi, V. Gabeur, Y.-T. Hu, R. Hu, C. Ryali, T. Ma, H. Khedr, R. Rädle, C. Rolland, L. Gustafson *et al.*, "Sam 2: Segment anything in images and videos," *arXiv preprint arXiv:2408.00714*, 2024.

[43] B. Wen, W. Yang, J. Kautz, and S. Birchfield, "Foundationpose: Unified 6d pose estimation and tracking of novel objects," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 17 868–17 879.