

# Closing the Communication Loop for Robotic Failures: Multi-Turn, Behavior-Tree-Grounded Explanations with Large Language Models

Parag Khanna<sup>1</sup>, Haoyun Zhou<sup>1</sup>, Elmira Yadollahi<sup>2</sup>, Iolanda Leite<sup>1</sup>, Christian Smith<sup>1</sup>

**Abstract**—Robot failures during collaborative tasks can frustrate users and reduce trust. To address this, we developed a failure communication module that combines large language models (LLMs) with Behavior Trees (BTs) to generate interactive, context-aware explanations for task failures. The module supports three key processes: (1) initial (high/medium/low) leveled explanations, (2) interactive clarifications for user follow-up questions, and (3) explicit verification of user actions to close the recovery loop. By leveraging the BT structure and persistent interaction history, it generates responsive, multi-turn explanations and reduces redundancy for repeated failures. We implemented and evaluated this module in real-time robotic pick-and-place tasks and conducted a user study with 33 participants across three high/medium/low explanation conditions. The user study showed that the module improved resolution rates for challenging failures and reduced resolution times for simpler failures, demonstrating the effectiveness of LLM-powered, BT-grounded explanations in human-robot collaboration (HRC).

## I. INTRODUCTION

Robots are increasingly deployed in collaborative roles with non-expert users across manufacturing, healthcare, and service robotics. While these advances extend the scope of robot applications, the complexity and unpredictability of real-world environments often lead to execution failures due to sensor limitations, unforeseen obstacles, or task complexity. Such failures not only hinder task completion but also risk eroding user trust, as prior work has shown that clear and contextual failure communication is essential for both recovery and sustained trust in human-robot collaboration [1].

Existing robot failure explanation systems predominantly rely on manually crafted template-based utterances [1], [2]. While templates can convey failure causes and user actions in controlled scenarios, they lack the contextual flexibility to respond to user queries, clarify ambiguities, or adapt explanations over time. In particular, template-based systems cannot maintain an awareness of the interaction history, repeatedly delivering the same static response regardless of prior exchanges, user expertise, or the recurrence of similar failures. This rigidity limits both understanding and recovery, especially as users encounter the same or related failures across multiple interactions [1], [3], [4], [5], [6].

Recent advances in Large Language Models (LLMs) offer a transformative pathway. LLMs can generate contextually tailored explanations and have the potential to clarify user queries [7]. However, left ungrounded, LLMs may generate verbose, inconsistent, or overly technical explanations that are ineffective for non-experts [4], [7], [5]. In this work, we

<sup>1</sup> Department of Robotics, Perception, and Learning (RPL), KTH Royal Institute of Technology, Sweden, <sup>2</sup> Lancaster University, UK. paragk@kth.se.

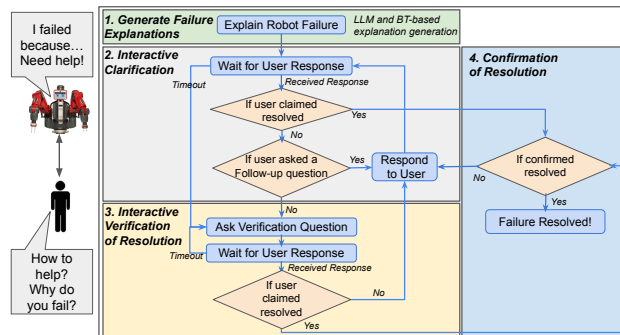


Fig. 1: Proposed Failure Communication Module architecture with LLM-BT-generated explanations and user-response addressal with interactive clarification and verification of successful resolution of failure in HRC. Codeset at [8].

address these challenges by grounding the LLM in the HRC task structure using Behavior Trees (BTs). BTs can provide a structured, modular, and interpretable representation of the robot’s task logic, possible failure points, and potential human-assisted recovery strategies and have been utilized in dialogue systems [7]. We present a failure communication module (Fig. 1) that leverages this structure along with a persistent memory of past interactions, tracking the robot’s failures, user responses, and clarification requests. This allows the system to deliver responsive, multi-turn explanations that are both *situated* and *tailored* to the user’s demonstrated understanding. The module supports three interactive processes: (1) initial explanation of failures at varying levels of detail, (2) follow-up clarification in response to user queries, and (3) explicit verification of recovery actions to close the failure-resolution loop. This approach not only improves interaction quality but also gives the user a sense that the robot “remembers” their past interactions, which can foster a more collaborative and engaging experience while improving communication efficiency during repeated failures. In contrast to rigid template-based systems or verbose generative models, our approach balances structure and flexibility, enabling robust, real-time explanations grounded in the task execution logic. Moreover, this LLM–BT approach scales effortlessly: adding a new failure only requires updating the BT, while templated systems need laborious case-by-case explanation design, making this approach essential for complex robots with many failure modes.

The key contribution of this paper is a **BT-grounded, LLM-based robot failure communication module** that integrates structured task knowledge and interaction history to produce context-aware, multi-level explanations during

HRI and supports **multi-turn, interactive dialogue** for user-driven clarification, recovery verification, and adaptive responses to repeated failures, demonstrating a shift beyond template-based systems toward scalable and robust HRI.

## II. BACKGROUND AND RELATED WORK

### A. Robot Failure Explanation

Prior research has established that both the content and delivery of robotic failure explanations significantly impact user understanding, trust and the likelihood of successful intervention [1]. Early approaches formalized failure explanation levels based on information content: *action-based* and *action-based with history* which focus on the robot's executed actions; and *contextual-based* and *contextual-based with history* [1], which incorporate environmental context and when available, the history of task execution to provide a more comprehensive account of failure causes. Also, increasing explanation detail generally improves user understanding but can also raise cognitive load and interaction time [9].

To capture environmental context, methods have employed semantic scene graphs to identify the most relevant objects and relationships [3], as well as causal Bayesian networks to model cause-effect structure and generate contrastive rationales, especially in manipulation tasks [10]. However, these approaches often require extensive hand-crafted knowledge and lack flexibility in interactive scenarios. Template-based explanations have been widely used to deliver concise, resolution-oriented explanations to non-experts. For instance, [2] proposed three templated explanation levels: Low, stating only the failed action; Medium, adding the cause of failure; and High, including both the cause and the last successful action. Each level was paired with a resolution statement that provided varying degrees of guidance to help a collaborating human resolve the failure. While effective in simple, repetitive settings, these strategies cannot answer follow-up questions, clarify ambiguities, or adapt explanations to the user's evolving needs and prior interactions. Crucially, they lack awareness of the interaction history, leading to redundant or inappropriate explanations when failures recur [2].

### B. Behavior Trees for Explanation

Behavior Trees (BTs) are a hierarchical, modular formalism widely used for modeling and controlling robot task execution [11]. This formalism allows systematic traversal to identify executed actions and failure points, making BTs a natural substrate for grounding explanations [7]. Represented in a compact, token-efficient format (e.g., YAML), this enriched BT can serve as a knowledge base for a LLM to generate grounded and context-aware explanations [4], [7]. This combination enables staged delivery, offering context first and resolution steps when appropriate, for real-time HRI that feels both natural and efficient [5].

### C. LLMs for Interactive Explanation

LLMs are generative neural models based on the transformer architecture, pre-trained on vast corpora of textual

data, and capable of coherent, context-sensitive natural language generation [12]. In robotics, LLMs have been applied to task planning [12], [13], control [14], and social communication [15]. Recent advances in LLMs have opened the door to more flexible and interactive failure explanation [16], [17], [5], where they generate natural language explanations from structured robot knowledge and support multi-turn dialogue. Frameworks such as REFLECT leverage LLMs to summarize robot experiences and generate failure explanations from hierarchical, event-based summaries [16]. [7] also integrate BTs with LLMs to enable generalizable failure explanations; however, only for single failure instances without further clarifications.

A key advantage of LLMs is to interpret structured robot knowledge (e.g., BTs) and environmental context to generate adaptive, interactive explanations. Unlike static template systems, they can respond to follow-up questions, refine clarity, and adjust content based on user expertise and past interactions. By remembering previous exchanges, LLM-based modules deliver *situated* and *personalized* explanations for recurring failures, reducing redundancy and enhancing user trust and engagement [16], [7], [18].

Thus, by integrating BTs with LLM-powered explanation modules, robots can provide concise, accurate, and context-aware explanations that adapt to user interactions and history, enabling user-centered robot failure communication.

## III. METHODS

We considered the HRC task from [2] with a workflow and setup shown in Fig. 2. In this task, the robot performs a pick-and-place task for various objects: for each object, detects it on the table, picks it up, carries it, and places it on a shelf (Please refer to the accompanying video and [2] for details). When the robot encounters a failure, it requires human assistance to resolve the issue. During such failures, the robot uses our proposed failure communication module to provide explanations to the human, responds to any follow-up questions from the human, and verifies whether the failure has been successfully resolved before proceeding.

### A. Failure Information in Behavior Trees

Given their hierarchical and modular structure, BTs are ideal for managing complex tasks and facilitating knowledge extraction by LLMs. Specifically, we aim to incorporate information about robot failures and their resolution steps inside the task BT. We consider the following failures for the pick-and-place task in our user study:

- **Pick Failure:** The robot fails to pick up the object from the table because its current orientation prevents it from fitting properly into the robot's gripper. To resolve this, the user should lift the object from the table and place it directly into the robot's gripper.
- **Carry Failure:** After picking the object, the robot fails to carry it to the shelf since it is too heavy for the robot's arm. To resolve this, the user should take the object from the robot's gripper and place it on the shelf.

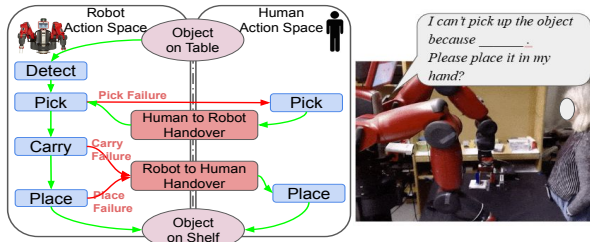


Fig. 2: Description and Setup of the HRC task for the user study adopted from [2], [6]. Transitions after a successful action are in green and those after a failed action are in red.

- **Place Failure:** After picking and carrying the object, the robot is unable to place it on the lower shelf since that area lies outside its reachable workspace. To resolve this, the user should take the object from the robot’s gripper and place it on the lower shelf.

The BT for the above-described collaborative task is shown in Fig. 3 which mentions robot actions as well as the failure resolutions. After each ‘RobotPickObject’, ‘RobotCarryObject’, and ‘RobotPlaceObject’, additional nodes are introduced to handle potential failures. Since the resolution step for a carry failure involves the user directly placing the object onto the shelf, the robot does not need to execute a separate place action afterward. Thus, the condition is set to true in the BT once the carry failure has been resolved. In this work, it is assumed that the causes and the corresponding resolutions of the robot failures are known in advance for specific actions with explicit encoding in the BT by augmenting the following information to the corresponding action nodes in the BT:

- **Action Description:** A description of the action the robot is performing.
- **Failure Cause:** A description of the potential causes that might lead the robot to fail in performing that action.
- **Resolution Step:** A description of the actions a human collaborator can do to resolve the failure for that action.

The LLM’s system prompt is provided with the complete behavior tree (BT), including failure-related information, represented as a YAML string (available in our Git repository: [8]). This structured format ensures that the information is expressed in rich natural language, forming the foundational knowledge base that provides the LLM with sufficient context to generate detailed explanations and respond to user follow-up questions. The YAML format is chosen because it

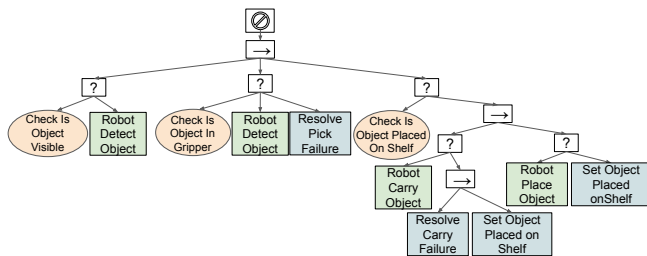


Fig. 3: The BT for the collaborative pick-and-place task with failure resolution nodes.

TABLE I: System and explanation prompts.

System Prompt
You are “Baxter”, a helpful robot collaborating with a human user on a task. The task is represented by the behavior tree below: “yaml {bt.yaml_string} ...”
This behavior tree is the only source of truth for the task. The human user’s speech will be transcribed into text by an ASR model. Some words may be misrecognized in the user’s speech, and you should correct them according to the context.
High-level Explanation Prompt
You failed to execute the action “{failed.action}”. Explain the failure to the user by:
1. Extract the last action before the current failed action from the behavior tree. 2. Explain to the user: In line 1, state that you have completed the last action, then state the current failure and failure cause. In line 2, state the resolution step. 3. Refine the explanation to be more concise while keeping all details for line 1 and line 2. Use first person pronouns in the explanation. Directly respond with the following format: Last action: <last action> Explanation: <line 1> <line 2> Refined explanation: <line 1> <line 2>
Changes in 1 and 2 for Prompt for Medium and Low level explanations
Medium-level
1. Explain to the user: In line 1, state the failure and failure cause. In line 2, frame a polite request for the user to do the resolution step, with only the action verbs and without all details of how to do the resolution step. The request should be phrased as a question. 2. Refine the explanation to be more concise by removing all unnecessary details. In line 2, state only the action verbs and without any specific details.
Low-level
1. Explain to the user: In line 1, state the failure without the cause. In line 2, state the resolution step with only the action verbs and without details. 2. Refine the explanation to be more concise by removing all unnecessary details.

is more token-efficient than JSON, offering a more compact representation of the BT structure for the LLM.

### B. Failure Communication Module

The failure communication module is the core component of our implemented HRI system, responsible for generating explanations and managing user interactions. Fig. 1 shows the architecture of this module, and the related codeset can be found at [8]. This module is built around an LLM that serves as an intermediary between the robot and the human user. It supports a multi-turn conversation between the LLM and the user, with the conversation history maintained within the LLM prompts to ensure contextual awareness throughout the interaction, enabling the LLM to generate contextually relevant responses informed by prior user queries and the robot’s previous explanations.

At initialization, the task knowledge base in the form of the BT YAML string (*bt.yaml\_string*) is provided by the system prompt to the LLM as shown in Table I.

During subsequent failure communication, the LLM is instructed by subsequent prompts to perform specific tasks, like generating explanations or responding to user queries, according to the current stage of the interaction. The failure communication module comprises four system-run processes: *Failure Explanation*, *Interactive Clarification*, *Interactive Verification of Resolution*, *Confirmation of Resolution*.

1) **Failure Explanation:** When a failure notification for a specific action, *failed\_action*, is received, *Failure Explanation* process is initiated by the system for providing the first explanation to the user post failure.

The content of the explanation is generated by the LLM as per the predefined explanation levels: high, medium and low,

TABLE II: Examples of the three levels of LLM-generated explanations for different failures in a pick-and-place task.

Task	Level	Failure Explanation	Resolution Step
Pick	High	I have detected the object, but I failed to pick it up because it does not fit in my gripper.	I have opened my gripper, please place the object in my gripper so I can grasp it.
	Medium	I failed to pick up the object because it does not fit in my gripper.	Could you pick up and hand over the object?
	Low	I failed to pick up the object.	Pick up and hand over the object.
Carry	High	I have picked up the object, but I failed to carry it to the shelf because it is too heavy for my arm.	Please take the object from me and place it on the shelf where you want it.
	Medium	I failed to carry the object because it is too heavy.	Could you take back and place the object?
	Low	I failed to carry the object.	Take back and place it.
Place	High	I have carried the object to the shelf, but I failed to place it because the destination is out of reach for my arm.	Please take the object from me and place it on the shelf where I cannot reach, so I can complete the task.
	Medium	I failed to place the object because the destination is out of reach.	Could you take back and place the object?
	Low	I failed to place the object.	Take back and place it.

defined in [2], essential for adaptive explanation verbosity in repeated failures [19]. The basic prompt to generate a high-level explanation, shown in Table I, includes the failed action’s name, instructs the LLM to extract the last successful action from the BT and asks the LLM to generate a draft explanation based on the failure information: the last successful action, the failed action description, the failure cause, and the resolution step. Because the failure information encoded in the BT (YAML [8]) includes detailed descriptions, the initial draft explanations generated by the LLM tend to be overly verbose. Thus, the LLM is additionally instructed to refine these drafts into concise, user-friendly explanations by removing unnecessary details.

A structured output format for the LLM response is also defined: the first line has the last successful action, the second has the draft explanation, and the third line has the refined explanation. Each explanation is formatted into two sub-lines: the failure context in the first sub-line and the resolution step in the second sub-line. This structured format improves the LLM’s output consistency as well as facilitates parsing during system execution. In real-time implementation, the refined explanation’s first line is delivered to the user immediately after the failure, while the second line (resolution step) is delivered when the robot is ready to accept the resolution step. For medium and low levels, the basic prompts have a similar structure to the high-level explanation prompt (Table I). The main differences are:

- The explanation content specified in the prompt varies depending on the explanation level [2]. The last successful action is removed from the medium and low level explanations. The failure cause is removed in the low level explanations.
- Each level includes tailored instructions to ensure that the generated resolution step aligns with the level used in prior work [2].
- Instructions for refining the explanation emphasize greater conciseness at lower explanation levels.

Table II shows examples of LLM-generated explanations for a pick failure, a carry failure, and a place failure in the pick-and-place task. These explanations were consistent across many trials, indicating the robustness and effective grounding of the LLM outputs with the underlying BT. Further, these closely align with the detail and clarity of template-based approaches reported in prior work [2], providing compara-

ble semantic content and appropriately varying information according to each explanation level.

Beyond the basic prompt structure, we additionally include extra instructions in the prompt in specific cases to further improve explanation generation. When a current failure repeats in a new task or involves a new object, we explicitly inform the LLM at the beginning of the prompt that ‘Now you are in a new task’ or ‘Now you are doing the task on a new object’. This prevents the LLM from assuming that the failure occurred in the same task or on the same object as previously discussed in the conversation history. For repeated failures where the same type of failure has occurred before, we instruct the LLM with ‘Take the history conversations into consideration when explaining’ to obtain a more history-aware explanation. For high-level explanations, this leads the LLM to explicitly inform the user that the failure has occurred previously, as shown in Table III. However, we observe this behavior less frequently in medium- and low-level explanations.

2) **Interactive Clarification.** Once the initial explanation of the failure is presented to the user, the system proceeds to this phase to handle possible user questions and support the failure resolution process. We classify user responses into 3:

- 1) The user claims that they have already completed the resolution step.
- 2) The user expresses uncertainty or asks for more details.
- 3) Other responses not covered by the first two categories. The most common cases are brief acknowledgments such as “Yes” or “Okay”, indicating understanding but not necessarily completion of the resolution step.

The system must be capable of distinguishing among these three cases and delivering appropriate responses. We utilize BT-LLM to tackle this challenge through a prompt for *Interactive Clarification*, as illustrated in Table IV (a), where *failed\_action* denotes the failed action and *user\_response*

TABLE III: Example of the high-level LLM-generated failure explanation for a repeated failure in a pick-and-place task.

Failure	Failure Explanation
Pick	I have detected the object, but I failed to pick it up because it does not fit in my gripper, like the first time.
Carry	I have picked up the object, but I failed to carry it to the shelf because it is too heavy for my arm, like before.
Place	I have carried the object to the shelf, but I failed to place it because the destination is out of reach for my arm, like before.

represents the transcript of the user’s spoken response.

First, the LLM is instructed to correct any misrecognized words in the user’s speech. This step is essential because the speech-to-text model used in real-time implementation may incorrectly recognize certain words. By fixing these errors, the LLM can more accurately interpret the user’s response and produce a reliable reply. The corrected text is required to appear on the first line of the output and is used for subsequent processing. This correction mechanism is also consistently applied in all later stages involving user speech.

Next, the LLM is reminded of the failed action and provided with the user’s response. It is then asked to determine: (1) whether the user has indicated completion of the resolution step, and (2) whether the user is confused or requesting additional details. If the answer to the first question is “Yes”, this indicates that the user claims to have completed the resolution step. The system then proceeds to the *Confirmation of Resolution* process (III-B.4) to verify whether the failure has actually been resolved. If the answer to the first question is “No” and the answer to the second question is “Yes”, the LLM generates a clarifying response that addresses the user’s concern and reiterates the resolution step. The system then remains in *Interactive Clarification* and awaits the user’s next response. If the answers to both questions are “No”, the system proceeds to the *Interactive Verification of Resolution* process (III-B.3).

**3) Interactive Verification of Resolution:** This process reminds the user and confirms whether the user has completed the resolution step in an interactive manner. The LLM first generates a concise, context-specific verification question focused on the last resolution step (e.g., “Have you placed the object in my gripper?” for a Pick Failure), using the prompt in Table IV(b).

Once the user responds, the LLM determines whether the response confirms completion of the resolution using the prompt in Table IV(c). If the user indicates that they have completed the resolution step, the system moves to the *Confirmation of Resolution* process (III-B.4) to verify actual resolution status. Otherwise, the LLM produces a response that addresses the user’s reply and reminds them to complete the resolution step. The system then returns to *Interactive Clarification* (III-B.2) waiting for user’s next response.

**4) Confirmation of Resolution:** In this process, the system verifies whether the failure has been resolved. In our user study, a Wizard-of-Oz human operator performs this check to ensure simplicity and accuracy. If resolved, the system proceeds with the next task action. If not resolved, it informs the user that the failure is not resolved and requests the resolution step again using an LLM-generated prompt as shown in Table IV (d). The system then returns to *Interactive Clarification* (III-B.2), waiting for the user’s next response.

**5) Timeout Mechanism and Maximum Number of Attempts:** To avoid indefinite waiting for user responses and give timely assistance when needed, a timeout of 3.5 seconds (based on pilot study observations) is set for explanation and verification prompts. If no response is received, the system transitions to *Interactive Verification of Resolution* stage and

TABLE IV: Various Interactive clarification and resolution verification prompts.

<p><b>(a) Prompt: Interactive Clarification</b></p> <p>After you explained the failure at "{failed_action}", the user said "{user_response}".</p> <ol style="list-style-type: none"> <li>1. There might be some misrecognized words in the user’s speech. Correct them according to the context and output the corrected text.</li> <li>2. Answer "Yes" or "No" to the question: Did the user indicate that they have already completed the resolution step?</li> <li>3. Answer "Yes" or "No" to the question: Is the user confused or asking for more details? If the answer is "Yes", also respond to the user’s statement and ask them to do the resolution step according to the context.</li> </ol> <p>Use first person pronouns. Do not include additional output. Output in the following format:</p> <p>&lt;Corrected text&gt; &lt;Yes/No&gt; &lt;Yes/No&gt; &lt;explanation (optional)&gt;</p>
<p><b>(b) Prompt: Verification Question Generation</b></p> <p>Ask the user the question according to the context: if they have completed the resolution step for "{failed_action}".</p> <p>Be concise. Only focus on the last resolution step. Directly output the question.</p>
<p><b>(c) Prompt: Resolution Verification</b></p> <p>You asked the user if they have completed the resolution step for "{failed_action}": The user replied "{user_response}" to your question.</p> <ol style="list-style-type: none"> <li>1. There might be some misrecognized words in the user’s speech. Correct them according to the context and output the corrected text.</li> <li>2. Answer "Yes" or "No" to the question: Did the user indicate that they have completed the resolution step?</li> </ol> <p>If the answer is "No", respond to the user’s statement and ask them to do the resolution step in the next line.</p> <p>Use first person pronouns. Do not include additional output. Output in the following format:</p> <p>&lt;Corrected text&gt; &lt;Yes/No&gt; &lt;response (optional)&gt;</p>
<p><b>(d) Prompt: Resolution Confirmation</b></p> <p>The user replied "{user_response}" after you asked them if they have completed the resolution step for "{failed_action}".</p> <p>However, the resolution step is not completed.</p> <p>According to the context, tell the detailed resolution step to the user and ask the user to carefully check it.</p> <p>Do not include "resolution step" in the response. Directly output the response.</p>

checks whether the user has completed the resolution step.

To avoid the user becoming trapped in a loop between *Interactive Clarification* and *Interactive Verification of Resolution*, the system restricts the number of attempts to obtain a user response to five, including the initial explanation and up to four follow-ups. This aligns with the maximum number of explanations provided for a robot failure in the prior user study [2]. Once this limit is reached, the system stops waiting for the user’s response, evaluates the resolution status of the failure, and proceeds to the next action in the task.

### C. Real-Time Implementation

The failure communication module was implemented in a real-time system for the user study. The *parakeet-tdt-0.6b-v2* model [20] from the *NeMo* framework [21] was used for speech recognition. For natural language generation the cloud-hosted *Gemini 2.0 Flash* model [22], which provided the fastest response time (comparison with other models in [8]). On average, it generates a 150-token response in 1.47 seconds. The robot’s speech responses were generated locally using the *Kokoro-82M* text-to-speech model [23].

## IV. USER STUDY

To evaluate the failure communication module, we used the same user study protocol as [6], [2]. We recruited 33 participants (23 male, 10 female; ages 22–44; M = 26.6, SD

= 4.9) without prior experience with collaborative robots. Participants were randomly assigned to three conditions:

- **E1 (Low):** Explanations always at low level.
- **E2 (Medium):** Explanations always at medium level.
- **E3 (High):** Explanations always at high level.

The fixed-level conditions (E1–E3) correspond to the prior template-based conditions C1–C3 in [2] (Fixed-Low, Fixed-Medium, Fixed-High), allowing direct comparison of LLM-powered explanations with template-based explanations in terms of resolution rate, resolution time, and user satisfaction across pick, carry, and place failures. Each participant performed four rounds of the collaborative task with a Baxter robot. The robot handled four objects per round with pre-programmed failures: pick, carry and place, as shown in Table V (refer to the accompanying video submission for details on failure actions and resolutions). When a failure occurred, the robot provided an LLM-generated explanation, moved its gripper to the handover position, and requested human assistance. This interaction continued until successful failure resolution or maximum repetition attempts were made. The participants rated their satisfaction on an 8-item, 5-point Likert scale ([24], [2]) covering understanding, completeness, utility, and trust after each round and in a final questionnaire that captured overall impressions. The performance was measured via:

- **Resolution Rate:** Proportion of failures successfully resolved.
- **Resolution Time:** Duration from robot’s initial explanation to confirmed resolution (in seconds).
- **User Satisfaction:** Average scores on the 8-item questionnaire. (available at our git: [8])

## V. RESULTS: IMPACT OF FAILURE COMM. MODULE

Firstly, it is important to note that the failure communication module produced stable first explanations for all three types of failures between all participants, consistent with the explanation levels and closely matching the templated explanation levels presented in [2]. This stability indicates that the BT-based grounding effectively supports the LLM’s capacity to generate accurate and contextually appropriate explanations at the desired levels. This module leads to a more natural interaction with the collaborating user compared to templated, repeated failure explanations from [2] and is expected to perform better in failure resolution and achieve better user satisfaction. Thus, we hypothesize that:

TABLE V: Objects and their action failures in the user study.

Round	Obj	Type	Failure	Round	Obj	Type	Failure
Round 1	A	Sponge	None	Round 2	A	Heavy Box 1	Carry
	B	Cloth Bag	Pick & Carry		B	Random Box 2	None
	C	Random Box 1	None		C	Flat Box	Pick
	D	Pen Box	Place		D	Toy	None
Round 3	A	Pen Box	Place	Round 4	A	Flat Box	Pick
	B	Heavy Pen Box	Carry		B	Sponge	None
	C	Toy	None		C	Pen Box	Place
	D	Random Box 3	None		D	Heavy Box 2	Pick & Carry

- **H1a:** The LLM-powered failure communication module (E1, E2, E3 conditions) achieves higher failure resolution rates compared to the template-based failure communication system (C1, C2, C3 conditions).
- **H1b:** This module (E1, E2, E3) achieves a shorter resolution time compared to the template-based (C1, C2, C3).
- **H1c:** This module (E1, E2, E3) achieves higher user satisfaction scores compared to the template-based system (C1, C2, C3).

The three pairs of conditions, E1 vs C1, E2 vs C2, and E3 vs C3, are compared to assess the effectiveness of our LLM-based failure communication module in III-B. To account for multiple comparisons, a Bonferroni correction was applied to the p-values, setting the significance threshold to  $\alpha = 0.05/3$ . Figs. 4, 5 show the resolution rate and resolution time variation across all interaction rounds and conditions.

### A. Resolution Rate

The resolution rates for all four rounds of failures are compared between conditions. Consistent with [2], all conditions achieve a 100% resolution rate for pick failures. Thus, we restrict the subsequent analysis to carry and place failures.

A chi-squared test indicated no significant differences in resolution rates between the LLM-powered and template-based explanations across all three condition pairs for carry failures: fixed-low (E1 vs C1,  $\chi^2(1) = 0.18$ ,  $p = 0.67$ ), fixed-medium (E2 vs C2,  $\chi^2(1) = 4.47$ ,  $p = 0.04$ ), and fixed-high (E3 vs C3,  $\chi^2(1) = 0.51$ ,  $p = 0.47$ ) conditions.

However, significant differences were shown by a chi-squared test in the resolution rates for place failures between the LLM-powered and template-based explanations for the fixed-low (E1 vs C1,  $\chi^2(1) = 7.92$ ,  $p = 0.005$ ) and fixed-medium (E2 vs C2,  $\chi^2(1) = 10.483$ ,  $p = 0.001$ ) conditions, but no significant difference for the fixed-high (E3 vs C3,  $\chi^2(1) = 2.21$ ,  $p = 0.13$ ) conditions.

Thus, for **H1a**, support for the hypothesis is observed only for place failures under the fixed-low and fixed-medium conditions. In contrast, the resolution rates for the carry failures are not significantly improved by the LLM-powered

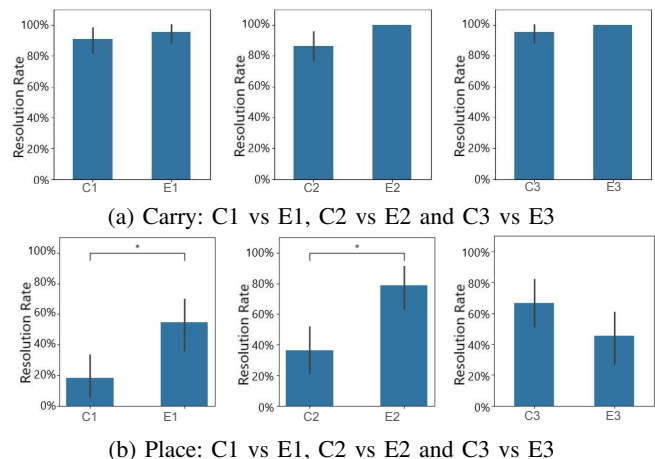


Fig. 4: Resolution rates for carry and place failures across all pairs of conditions. (E1 Low, E2 Medium, E3 High)

explanation module compared to the template-based explanations for all three condition pairs.

### B. Resolution Time

We analyze the resolution times (in seconds) for successful resolutions across all four failure rounds. For pick failures, the Shapiro-Wilk test indicated non-normal distributions for the resolution times in all conditions ( $p < 0.001$ ). Thus, the non-parametric Mann-Whitney U test was used to compare the resolution times between the LLM-based and template-based explanations, with the former yielding significantly shorter resolution times across all three condition pairs: fixed-low (E1 vs C1,  $U = 1559.0$ ,  $p < 0.001$ ), fixed-medium (E2 vs C2,  $U = 1647.0$ ,  $p < 0.001$ ), and fixed-high (E3 vs C3,  $U = 1365.0$ ,  $p < 0.001$ ).

Among carry failures, only C1 condition had normally distributed resolution times (Shapiro-Wilk test,  $p = 0.054$ ). Thus, the Mann-Whitney U test was used again, indicating that the resolution times were significantly shorter for the LLM-powered explanations than the template-based ones for fixed-medium conditions ( $U = 1314.0$ ,  $p = 0.004$ ). There were no significant differences for the fixed-low ( $U = 1190.0$ ,  $p = 0.065$ ) and fixed-high conditions ( $U = 1242.0$ ,  $p = 0.022$ ).

For place failures as well, only C1 condition had normally distributed resolution times (Shapiro-Wilk test,  $p = 0.151$ ). The Mann-Whitney U test indicated that the LLM-powered explanations lead to significantly longer resolution times than the template-based ones for the fixed-low ( $U = 337.0$ ,  $p = 0.008$ ) and the fixed-high conditions ( $U = 303.0$ ,  $p = 0.002$ ), while no significant differences were seen for the fixed-medium conditions ( $U = 654.0$ ,  $p = 0.162$ ).

Thus, for **H1b**, our results show that LLM-powered explanations generally did not produce a significant reduction in resolution time. The hypothesis is supported for pick failures across all three explanation levels and for carry

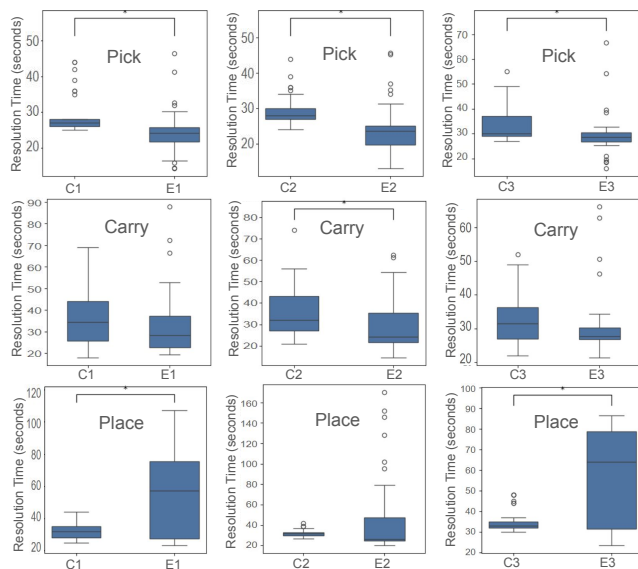


Fig. 5: Resolution times for pick, carry, and place failures across all condition-pairs. (E1 Low, E2 Medium, E3 High)

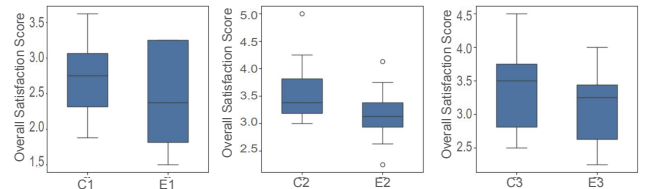


Fig. 6: Overall satisfaction scores in the 3 condition-pairs.

failures under the fixed-medium condition. In contrast, for place failures, the LLM-powered explanations resulted in significantly longer resolution times in the fixed-low and fixed-high conditions, contrary to our initial expectations.

### C. User Satisfaction

We begin by assessing the internal consistency of the user satisfaction questionnaire using Cronbach's  $\alpha$ , which indicated a high level of reliability. To evaluate the effect of the LLM-powered failure communication module on user satisfaction, we examine the overall satisfaction scores obtained from the final questionnaire, shown in Fig. 6. The Shapiro-Wilk test indicated that user satisfaction scores followed a normal distribution for all conditions except C2 ( $p = 0.031$ ) and E1 ( $p = 0.044$ ). Hence, the non-parametric Mann-Whitney U test was used for the fixed-low and fixed-medium conditions, while an independent t-test was applied for the fixed-high conditions. There were no significant differences in user satisfaction scores between the LLM-powered and template-based explanations: fixed-low ( $U = 71.0$ ,  $p = 0.509$ ) and fixed-medium ( $U = 86.5$ ,  $p = 0.091$ ) conditions and fixed-high conditions ( $t(20) = 1.001$ ,  $p = 0.329$ ). Overall, the LLM-powered module did not lead to higher user satisfaction compared to the template-based explanations in any of the three fixed-level conditions, providing no support for **H1c**.

## VI. DISCUSSION

The LLM-powered BT-grounded communication module is aimed at interactive, multi-turn explanations for robotic task failures by (1) explanation generation at different levels, (2) interactive clarification, and (3) verification of user recovery actions. Our evaluation demonstrates both the promise and complexities of integrating LLMs into real-time HRC.

The module produced stable, contextually appropriate explanations across all failure types and explanation levels, with consistency comparable to prior template-based approach [2] for all participants. This stability validates the robustness of the system and supports the effectiveness of grounding the LLM outputs within the BT representation of the collaborative task to maintain coherent, task-relevant outputs while enabling flexibility. The module's ability to reference interaction history and adapt explanations for repeated failures represents a significant advance over rigid template systems. Unlike template-based systems that deliver identical responses regardless of context, our module demonstrated awareness of previous exchanges, reducing redundancy and personalizing communication based on demonstrated user understanding. The module's primary strength lies in its scalability, adding new failure types requires only updating

TABLE VI: Example user-robot conversations assisted by the LLM-powered failure communication module.

Failure Type	Role	Speech
Carry Failure	Robot	I failed to carry the object.
	Robot	Take back and place the object.
	User	Take it back and place it back in your grip, you mean?
	Robot	No, I mean you should take the object back from my gripper and place it on the shelf.
	Robot	
Place Failure	Robot	I failed to place the object because the destination is out of reach.
	Robot	Could you take back and place the object?
	User	Yeah.
	Robot	Have you placed the object?
	User	Yes, I have.
	Robot	I understand that you have placed the object. Could you please check if you have placed the object at the correct location?
User	Oh! the correct location. Now, I have checked and it is placed at the correct location.	

the BT representation, while template-based systems need extensive manual crafting of explanations for each failure type. This advantage becomes crucial for complex robotic systems with numerous potential failure scenarios.

Via our user study, we find that the failure communication module significantly improved the resolution rate for challenging failures (place failures) compared to template-based explanations, while reducing resolution time for simpler failures (pick and carry) through clearer instructions. However, challenging failures showed increased resolution time due to longer, more detailed explanations that help users understand complex issues but require additional processing time (as in Table VI). This creates a trade-off between explanation thoroughness and interaction speed: our module’s richer, context-laden explanations with clarifications improve resolution rates by supplying necessary contextual cues where static templates fail, but extend interaction time due to added cognitive load. Since our timing metrics include only successful resolutions, template-based explanations may appear faster but actually fail more often. A true comparison must include both recovery rate and completion time, as the longer times likely reflect cognitive overhead from parsing richer context that ultimately drives successful completion.

A limitation of this study was that the module lacked full awareness of the environment and user actions, which can introduce minor ambiguities in the explanations, leading to confusion. Thus, future improvements could integrate Vision Language Models (VLMs) for visual context and additional user behavior features to enhance accuracy and reliability.

## VII. CONCLUSIONS

We developed an LLM-BT-powered failure communication module that provides interactive, context-aware explanations to assist users in resolving robot failures, interactively addresses user follow-up questions and verifies failure resolution. The evaluative user study shows that the module improves resolution rate for difficult failures and reduces resolution time for simpler failures, while detailed explanations for complex failures may increase resolution time. For future work, we will evaluate this module in different HRC and HRI tasks. We will also incorporate VLMs for visual context and for replacing the Wizard-of-Oz for resolution verification, along with richer user behavior features to improve the accuracy and consistency of explanations, adapting

the explanations to reduce confusion and enhance user trust in human-robot collaboration.

## REFERENCES

- [1] D. Das, S. Banerjee, and S. Chernova, “Explainable ai for robot failures: Generating explanations that improve user assistance in fault recovery,” in *ACM/IEEE Int. Conf. on Human-Robot interaction*, 2021.
- [2] P. Khanna, E. Yadollahi, M. Björkman, I. Leite, and C. Smith, “Effects of explanation strategies to resolve failures in human-robot collaboration,” in *IEEE Int. Conf. on Robot and Human Interactive Communication (RO-MAN)*, 2023, pp. 1829–1836.
- [3] D. Das and S. Chernova, “Semantic-based explainable ai: Leveraging semantic scene graphs and pairwise ranking to explain robot failures,” in *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2021.
- [4] 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 Transactions on Human-Robot Interaction (THRI)*, vol. 10, no. 3, pp. 1–31, 2021.
- [5] G. LeMasurier, C. Tagliamonte, J. Breen, D. Maccaline, and H. A. Yanco, “Templated vs. generative: Explaining robot failures,” in *IEEE RO-MAN*, 2024, pp. 1346–1353.
- [6] P. Khanna, E. Yadollahi, M. Björkman, I. Leite, and C. Smith, “User study exploring the role of explanation of failures by robots in human robot collaboration tasks,” *arXiv preprint arXiv:2303.16010*, 2023.
- [7] C. Tagliamonte, D. Maccaline, G. LeMasurier, and H. A. Yanco, “A generalizable architecture for explaining robot failures using behavior trees and large language models,” in *ACM/IEEE HRI*, 2024.
- [8] Parag Khanna and Haoyun Zhou, “Repository with code-set, bt-yaml and further information,” 2026, accessed: 2026-03-06. [Online]. Available: [https://github.com/paragkhanna/LLM\\_BT\\_Explanations](https://github.com/paragkhanna/LLM_BT_Explanations)
- [9] R. Linder, S. Mohseni, F. Yang, S. K. Pentylala, E. D. Ragan, and X. B. Hu, “How level of explanation detail affects human performance in interpretable intelligent systems: A study on explainable fact checking,” *Applied AI Letters*, vol. 2, no. 4, p. e49, 2021.
- [10] M. Diehl and K. Ramirez-Amaro, “Why did i fail? a causal-based method to find explanations for robot failures,” *IEEE Robotics and Automation Letters*, vol. 7, no. 4, pp. 8925–8932, 2022.
- [11] M. Colledanchise and P. Ögren, *Behavior trees in robotics and AI: An introduction*. CRC Press, 2018.
- [12] W. Huang, P. Abbeel, D. Pathak, and I. Mordatch, “Language models as zero-shot planners: Extracting actionable knowledge for embodied agents,” in *Int. Conf. on Machine Learning*, 2022, pp. 9118–9147.
- [13] C. H. Song, J. Wu, C. Washington, B. M. Sadler, W.-L. Chao, and Y. Su, “Llm-planner: Few-shot grounded planning for embodied agents with large language models,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 2998–3009.
- [14] Y.-J. Wang, B. Zhang, J. Chen, and K. Sreenath, “Prompt a robot to walk with large language models,” *arXiv:2309.09969*, 2023.
- [15] J. Sevilla-Salcedo, E. Fernández-Rodicio, L. Martín-Galván, Á. Castro-González, J. C. Castillo, and M. A. Salichs, “Using large language models to shape social robots’ speech,” *International Journal of Interactive Multimedia and Artificial Intelligence*, 2023.
- [16] Z. Liu, A. Bahety, and S. Song, “Reflect: Summarizing robot experiences for failure explanation and correction,” *arXiv preprint arXiv:2306.15724*, 2023.
- [17] J. Duan, W. Pumacay, N. Kumar, Y. R. Wang, S. Tian, W. Yuan, R. Krishna, D. Fox, A. Mandlekar, and Y. Guo, “Aha: A vision-language-model for detecting and reasoning over failures in robotic manipulation,” *arXiv preprint arXiv:2410.00371*, 2024.
- [18] P. Khanna, A. Naoum, E. Yadollahi, M. Björkman, and C. Smith, “REFLEX Dataset: A Multimodal Dataset of Human Reactions to Robot Failures and Explanations,” in *ACM/IEEE HRI*, 2025, pp. 1032–1036.
- [19] A. Naoum, P. Khanna, E. Yadollahi, M. Björkman, and C. Smith, “Adapting robot’s explanation for failures based on observed human behavior in human-robot collaboration,” in *IEEE/RSJ IROS*, 2025.
- [20] Nvidia, “Parakeet tdt 0.6b v2,” <https://huggingface.co/nvidia/parakeet-tdt-0.6b-v2>, 2025.
- [21] —, “Nvidia nemo framework,” [github.com/NVIDIA/NeMo](https://github.com/NVIDIA/NeMo), 2025.
- [22] G. Cloud, “Gemini 2.0 flash,” <https://cloud.google.com/vertex-ai/generative-ai/docs/models/gemini/2-0-flash>, 2025.
- [23] Hexgrad, “Kokoro-82m,” [huggingface.co/hexgrad/Kokoro-82M](https://huggingface.co/hexgrad/Kokoro-82M), 2024.
- [24] R. R. Hoffman, S. T. Mueller, G. Klein, and J. Litman, “Metrics for explainable ai: Challenges and prospects,” *arXiv preprint arXiv:1812.04608*, 2018.