

1 A Dataset Details

2 A.1 Robot Tasks and Implementations

3 We automated four mobile manipulation tasks in a home environment: putting a cup in the sink,
4 microwaving food, hanging a hat, and collecting dirty clothes. The implemented tasks made up of
5 autonomous navigation, perception, and manipulation actions. We navigate to specified positions
6 and orientations using SLAM implemented in Stretch Nav2, detect and localize specified objects by
7 either running ArUco marker detection or YOLO-World object detection with FastSAM segmenta-
8 tion, and perform manipulation using inverse kinematics with demonstrated poses. The task-specific
9 descriptions and implementations are described below:

- 10 • **Put Cup in Sink:** The task is performed in a connected kitchen and lounge environment
11 with a dirty cup in the lounge and a sink in the kitchen. The specific sequence of states
12 executed by the robot is as follows: the robot navigates to a table, looks for a cup, picks up
13 a cup from the table, navigates to the sink, looks for the sink, then places it in a sink.
- 14 • **Microwave Food:** The task is performed in a kitchen environment with a microwave and
15 food near the microwave. The specific sequence of states executed by the robot is as fol-
16 lows: the robot navigates to the microwave, looks for the microwave, opens the microwave
17 door, navigates to the food, looks for the food, picks up the food, navigates to the mi-
18 crowave, looks for the microwave, places the food inside the microwave, then closes the
19 microwave door.
- 20 • **Hang Hat:** The task is performed in a lounge environment with a human wearing a hat
21 and a hook. The specific sequence of states executed by the robot is as follows: the robot
22 navigates to the human, is handed a hat from the human, navigates to the hook, looks for
23 the hook, then hangs the hat on the hook.
- 24 • **Collect Dirty Clothes:** The task is performed in a lounge environment with a laundry
25 basket and clothes arranged around the room. The specific sequence of states executed by
26 the robot is as follows: the robot navigates to the clothes, looks around for clothes, classifies
27 dirty clothes, picks up dirty clothes, navigates to the laundry basket, looks for the laundry
28 basket, then places the clothes in the laundry basket.



Figure 1: The robot detects and localizes the cup using YOLO-World and segments the result with FastSAM on a frame captured from its head camera for perception.

29 A.1.1 State Machine Synthesis

30 We represent sequences of actions that accomplish a subgoal of a task as a state, and manage transi-
 31 tions between states using state machines. To simplify the development process, we create config-
 32 urable templates to synthesize state machine code from a list of linear states. In each implemented
 33 task, we add in additional query user and teleoperation states on top of each existing action state.
 34 This allows us to manage the state of the robot after the robot experiences failures and teleoperate
 35 the robot to recover from failures.

36 A.2 Dataset Overview

Task	Trials	Failure Counts		
		Navigation	Detection	Manipulation
Put Cup In Sink	32	10	2	18
Microwave Food	17	2	0	22
Hang Hat	16	0	0	14
Collect Dirty Clothes	5	0	5	0
Total	70	12	7	54

Table 1: List of autonomous tasks implemented on the Stretch mobile manipulator, along with the number of trials and failure counts separated by failure type collected for the dataset.

37 Tasks are run across different trials. Throughout each trial, whenever the robot encounters an inci-
 38 dent that prevents the robot from completing its task, a failure is marked. If the failure is recoverable,
 39 a human will teleoperate the robot to resolve the failure. Otherwise, the task is aborted and marked
 40 accordingly. Room arrangements are changed to inject failures into trials. Each trial can have from
 41 zero to three failures. The tasks and failure counts are shown in Table 1.

42 A.3 Data Collection

43 The data is collected as a rosbag that has RGB-D, joint, odometry, state information, and more
 44 beginning at the start of the task and ending after the task is complete. The full list of collected
 45 topics and topic descriptions are provided in 2.

Topic Name	Description
/state_machine/smach/container_status	Current state status information
/state_machine/smach/container_structure	State machine container information
/camera/aligned_depth_to_color/image_raw	Head mounted D435i stereo depth
/camera/color/image_raw	Head mounted D435i RGB image
/gripper_camera/color/image_rect_raw	Gripper mounted D405 RGB image
/camera/aligned_depth_to_color/camera_info	Head mounted D435i camera intrinsics
/stretch/joint_states	Joint positions, velocities, and efforts
/odom	Base odometry information
/imu_mobile_base	Base mounted IMU information
/tf	Transforms for moving robot parts
/tf_static	Transforms for static robot parts
/robot_description	The robot’s URDF
/rosout	ROS console logs
/diagnostics	Diagnostics information
/amcl_pose	The robot’s Nav2 pose with covariance

Table 2: List of all of the ROS topics and topic descriptions collected in each trial of the dataset.

46 B Implementation Details

47 This section introduces the implementation details about the framework which is not demonstrated
48 in the main paper due to page limit.

49 B.1 Multi-Modal Key Event Selection

50 B.1.1 A Formalism for Key Event Selection

51 We define a key event as the event which contains important and critical information on robot ob-
52 servations and robot status during task execution. These events are selected across all the modalities
53 of the robot. Since key events can have different definitions depending on the modalities, based on
54 the properties of the data, we categorize robot data into three categories: *Environment (E)*, *Internal*
55 *(I)* and *Task Planning (TP)*:

- 56 • **Environment (E)**: Sensory data used to observe the external world, such as RGB images,
57 point clouds, audio, tactile feedback, etc.
- 58 • **Internal (I)**: Sensory data related to the internal state of the robot, including internal sen-
59 sors, joint angles, base velocity, battery levels, and other diagnostic information.
- 60 • **Task Planning (TP)**: High-level planning data that contains overall task objectives, sub-
61 task sequences, execution history, and plan outcomes.

62 These categorizations can group the robot data in a structured way and make key event selection
63 across modalities easier. Then, we represent each aligned multi-modal frame with the following
64 parameters across different data categories:

- 65 • **Optical Flow (E)**: the optical flows generated from the RGB images collected by the robot
66 head camera. Since the optical flow vary largely at different stages of task operation, we de-
67 fine average flow magnitudes on different part movements of the robot. It has the following
68 *parameters*:
 - 69 – λ^{pos} : Average flow magnitude for frames with base positional movements.
 - 70 – λ^{rot} : Average flow magnitude for frames with base rotational movements.
 - 71 – λ^{cam} : Average flow magnitude for frames with camera movements.
 - 72 – λ^{arm} : Average flow magnitude for frames with arm positional movements.
- 73 • **Joint State (I)**: the internal joint state readings from the robot which reflects the robot
74 internal status. It has following *parameters*:
 - 75 – x^{pos} : Change in meters of the position of the robot from the odometry reading.
 - 76 – x^{rot} : Change in radians of the orientation of the robot from the odometry reading.
 - 77 – x^{cam} : Change in radians of the pan and tilt of the robot’s head camera.
 - 78 – x^{arm} : Change in meters of the robot’s arm.
- 79 • **Planner State (TP)**: the planning-level state information of the robot during the task exe-
80 cution. It has following *parameter*:
 - 81 – s : The robot’s current state.

82 For key event selection, we first calculate the mean and standard deviation of each parameter in
83 optical flow and joint state across each task. Then, for each multi-modal frame, we normalize each of
84 these parameter values by the mean and standard deviation of the specified task. We only take multi-
85 modal frames with normalized values above 0 as the candidates of key events. The normalization on
86 environment and internal parameters ensures values across different modalities are weighted equally,
87 and that only values that are higher than the average frame value will be considered for contributions
88 to the set threshold and further to become a key event. These values are accumulated along with the
89 task execution and a set threshold is set for the cumulative sum. Additional to optical flow and joint

state, planner state is used separately for key event selection. As in task planner, the most important events are the beginning and finishing of each planner state, and therefore, these events should be also considered as key events. In summary, when either the cumulative sum of values reaches the set threshold or the planner state changes, the multi-modal frame is labeled as key event. For our data, we aligned frames with a sample rate of 0.2 and set our key event threshold to 80. The key event selection can be modeled as a binary classifier, C_{key} , which runs across all the multi-modal frames in a given task and outputs a binary prediction, 0 (not a key event) or 1 (a key event). It can be represented as following:

$$C_{key}(f_i) = \begin{cases} 1, & \sum_{k=c}^i \left(\sum_{a \in \{\text{pos, rot, cam, arm}\}} (\mathcal{N}(\lambda_k^a) + \mathcal{N}(x_k^a)) \right) > \text{threshold}, \\ & \text{where } c = \text{index of last key event} \\ 1, & s_i \neq s_{i-1} \\ 0, & \text{otherwise} \end{cases} \quad (1)$$

where f_i is the multi-modal frame at timestamp i .

B.1.2 Adjacent Image Selection with Clarity Score

After selecting the key event, one additional step in our framework is to select the best quality RGB image adjacent to the key event. During the experiments, we found that there is a big impact on the clarity of the image for object detection, especially during navigation. Therefore, in order to improve the detection accuracy and further enhance the overall system, we define a clarity score on the RGB images and conduct an adjacent image selection. The clarity score is defined by the variance of the Laplacian of the image. The 2-D Laplacian operator is defined as:

$$\nabla^2 f = [f(x+1, y) + f(x-1, y) + f(x, y+1) + f(x, y-1)] - 4f(x, y) \quad (2)$$

We can use the equation to derive a 3×3 Laplacian kernel. Then, we use the kernel to convolve with the original image to get the activation map. By calculating the variance of the activation map, we can get a clarity score of the image. The higher the score is, the more clear the image is. We use the clarity score function to calculate clarity scores for all the frames within 1 second of the key event and select the RGB image with the highest clarity score as the image used for experience summarization.

B.2 RONAR-UI

To create better user experiences, we also design an user interface for the key event and narration display, RONAR-UI. The RONAR-UI has two modes, offline mode and online mode. The offline mode is used for users to log and analyze the collected robot data. Additionally, it is also used for user studies in C.2. The online mode is used for robot operators to lively monitor and control the robot. It integrates with

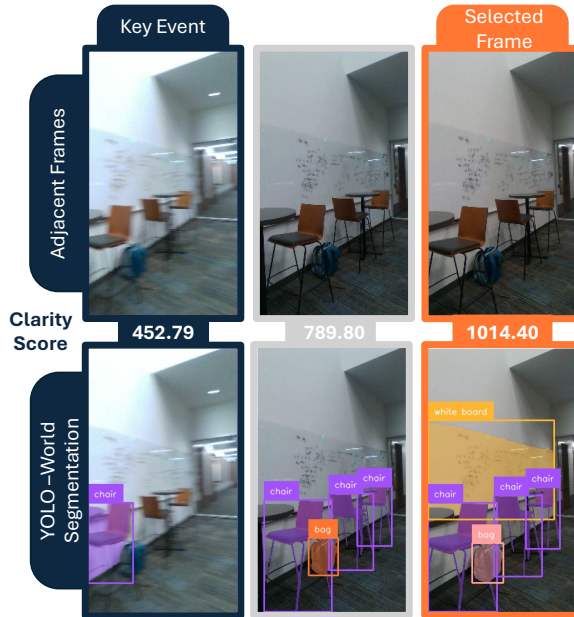


Figure 2: Adjacent image selection using clarity score. The left dark blue frame is the key event corresponding RGB image. The right orange frame is the selected frame used for experience summarization by using clarity score.

130 ROS2 with a user-friendly interface to allow users to learn the status of the robot system and do
 131 interventions during task execution. The RONAR-UI interfaces are shown in Figure 3.

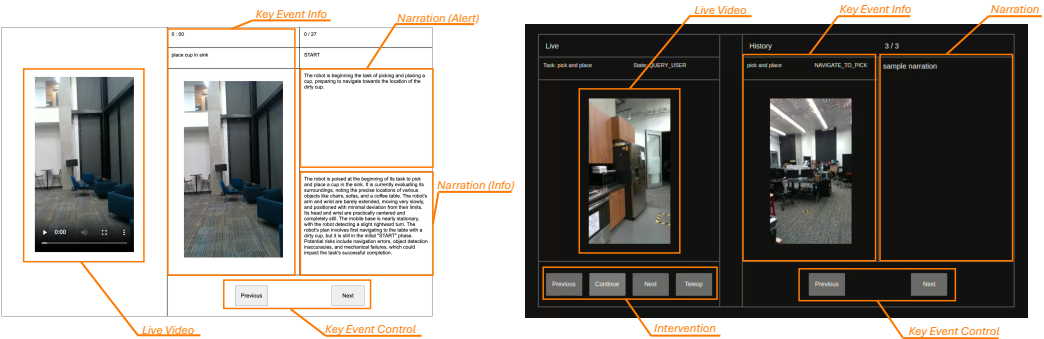


Figure 3: RONAR-UI interfaces. **Left:** RONAR-UI with offline mode. **Right:** RONAR-UI with online mode.

C Experiment Details

RONAR can generate multi-modal experience summaries and progressive narrations with different modes. In order to prove the effectiveness of these generated natural language groundings, we design multiple experiments. There are two main parts we want to prove RONAR can help with:

- **Robot System:** RONAR can improve the capability of robot systems on failure analysis.
- **Human Interaction:** RONAR can improve the interaction experience between human and robot. As well, RONAR can enhance the effectiveness and efficiency of failure identification by human users.

In order to prove RONAR can improve the robot system, we design an experiment on failure analysis by using generated experience summaries and you can find details in C.1. To prove the quality, effectiveness and efficiency of the narrations, we design two user studies shown in C.2. In C.2.2, we introduce the details of the user study on the narration quality evaluation. In C.2.3, we introduce the details of the user study on users’ effectiveness and efficiency on failure identification by using narrations.

C.1 Failure Analysis with Experience Summary

In order to make intelligent robots, it is important for robot systems to have reliable and efficient methods to identify and analyze failures. In this experiment, we aim to demonstrate that RONAR’s generated experience summaries can significantly enhance the failure analysis capabilities of robot systems. In order to have a thorough analysis, we break down the failure analysis problem into four sub-tasks:

- **Risk Estimation / Failure Prediction (Pred):** given previous key events, percentage of predicted failures are the actual failure in the actual failure key event.
- **Failure Localization (Loc):** given the all key events, percentage of predicted failure time are aligned with the ground truth failure time.
- **Failure Explanation (Exp):** given previous key events and current key event (when failure happened), percentage of generated failure explanations are aligned with the ground truth failure explanation.
- **Recovery Recommendation (Rec):** given previous key events and current key event (when failure happened), percentage of recovery recommendation are aligned with the ground truth recovery recommendation.

These sub-tasks include most scenarios the robot systems need to face during the operations. The methods used for comparison are:

- **BLIP2:** use BLIP2 to generate a caption for the RGB image of the key event.
- **REFLECT:** current state of the art LLM-based failure explanation framework.
- **TEM-LLM:** send all raw sensory and planning data directly to the LLM for failure analysis.
- **TEM-VLM:** send all raw sensory and planning data directly to the VLM for failure analysis. We use GPT-4o as VLM.
- **RONAR-vision_only:** our method without internal and planning inputs.
- **RONAR-no_prior:** our method only uses current key events for failure analysis.

Baseline methods, which require an LLM, use GPT-4o as the backbone. For REFLECT, we use the general pipeline of the method, but we did not use the audio modality since the original robot does not have microphone which might have potential effects on the performance of the method on some tasks. The results are evaluated by human expert based on the reasonableness and deviation from the ground truth labels.

C.2 User Studies

In order to prove the quality, effectiveness and efficiency of RONAR narrations, we carefully design two experiments, narration quality evaluation and failure identification using narration, and recruit a number of participants to conduct a thorough user study.

C.2.1 Demographics and Robot Background of Participants

We recruited 24 participants with different background to conduct the user studies on narration quality evaluation and failure identification using narration. We create a questionnaires to ask for participants' background at the end of the user study. The questionnaires include two parts: demographics and robot background. For the demographics, we include the following questions:

- **Age:** Select your age range (under 18, 18-24, 25-35, 35-44, 45-54, 55-64, over 65)
- **Education:** What is your highest level of education? (High school diploma / GED, Associate degree, Bachelor's degree, Master's degree, Doctorate)
- **Filed:** Have you studied or worked in a tech or STEM related field? (Yes or No)

The details of participant demographics can be found in Figure 4. From Figure 4, we can see that the user study involved a predominantly young and highly educated group of participants. The age distribution shows a significant concentration in the 18-24 age range, while the educational background highlights that most participants hold at least a Bachelor's degree, with nearly half having attained a Master's degree. This demographic profile suggests that the findings of the study might be particularly relevant to younger, well-educated individuals.

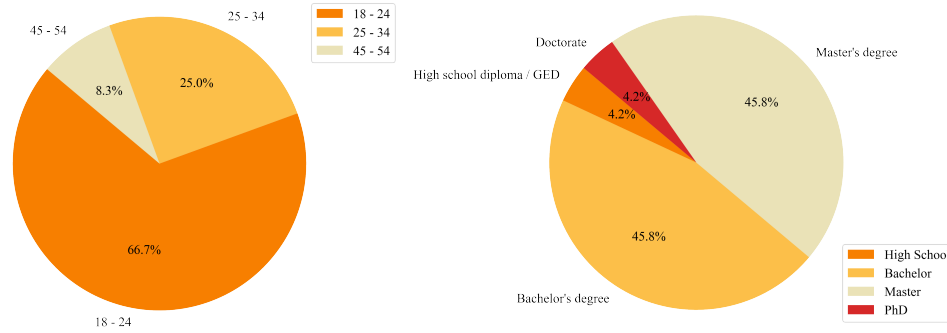


Figure 4: Demographics of the participants. **Left:** Age ranges of the participants of the user study. **Right:** Highest degrees earned by the participants of the user study.

We also create questions on robot familiarity for participants to answer. These questions include:

- **Expertise Level (Robot):** Rate your expertise in robotics (1-5)
- **Hours Spent (Robot):** Estimate how many hours have you worked with real robot? (Never, 0-10h, 10-30h, 30-50h, 50-100h, More than 100h)
- **Expertise Level (Stretch):** Rate your expertise with Hello Robot's Stretch mobile manipulator (1-5)
- **Hours Spent (Stretch):** Estimate how many hours have you worked with stretch robot? (Never, 0-10h, 10-30h, 30-50h, 50-100h, More than 100h)

The details of participant expertise on robots can be found in Figure 5. From Figure 5, it is clear that while the participant pool is quite diverse in terms of general robotics experience, they predominantly lack familiarity with the Stretch robot. Despite a few individuals with substantial robotic experience, the overall expertise with Stretch is low. Furthermore, the distribution of expertise level

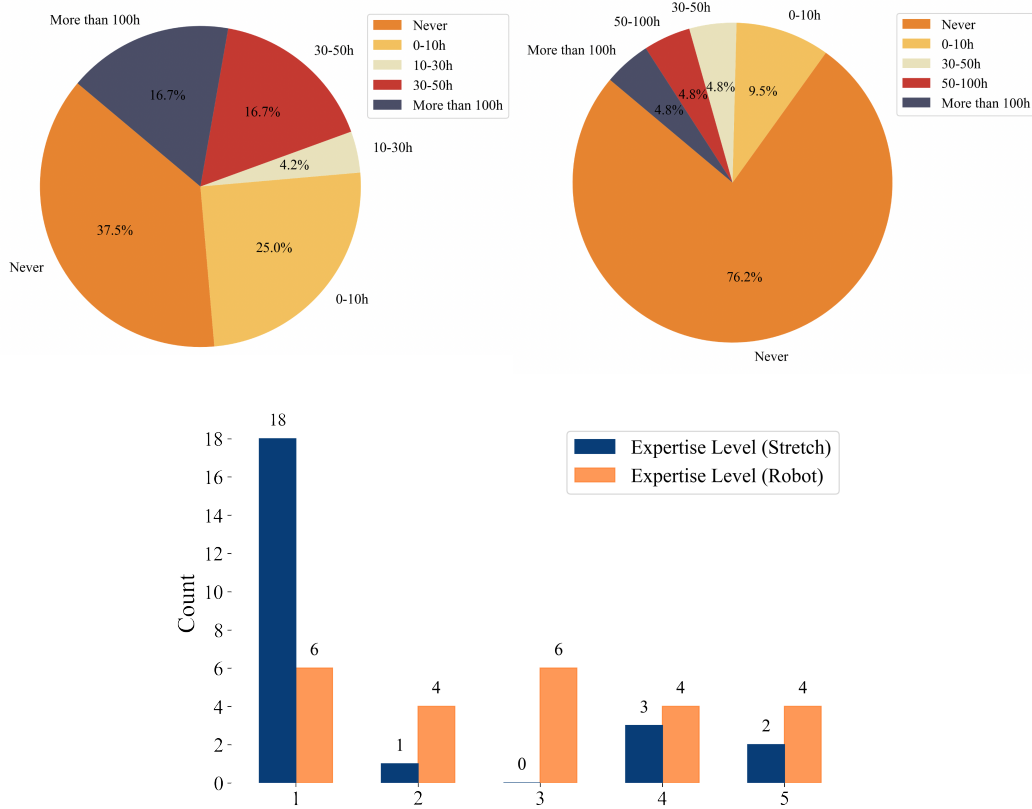


Figure 5: Expertise of the participants with robot and Stretch. **Top Left:** Hours spent by the participants on robots in general. **Top Right:** Hours spent by the participants on Stretch. **Bottom:** Subjective self-evaluation on the expertise level of the participants on general robot and Stretch.

on general robots shows a relatively even spread of expertise levels, with a notable concentration at both the novice and intermediate levels, reflecting a varied participant pool in terms of general robotics knowledge.

C.2.2 Narration Quality Evaluation

The study of narration quality evaluation has two parts: a tutorial and a formal evaluation. Before the start of the tutorial, we introduce the four metrics the participants give ratings on:

- **Naturalness:** Does the narration feel natural and human-like? (1-5)
- **Informativeness:** Does the narration provide useful information about the robot’s behavior? (1-5)
- **Coherence:** Does the narration organize information logically and clearly? (1-5)
- **Overall:** What is your overall assessment of the narration’s quality? (1-5)

It uses 1 to 5 Likert Scale to measure the preferences from the participants on the metrics. We confirm and explain the metrics until the participants fully understand the task and the terminologies. Then, the participants are given a short tutorial on two samples of the image-narration pairs to get familiar with the format and questions. They can ask any related questions during the tutorial. After the tutorial, it goes to the formal evaluation of the narrations generated by different methods. In the formal evaluation, we select three frames from three tasks in the dataset: put cup in sink, microwave lunch and hang hat. There are 5 methods evaluated by the participants: BLIP2, REFECLET, TEM-

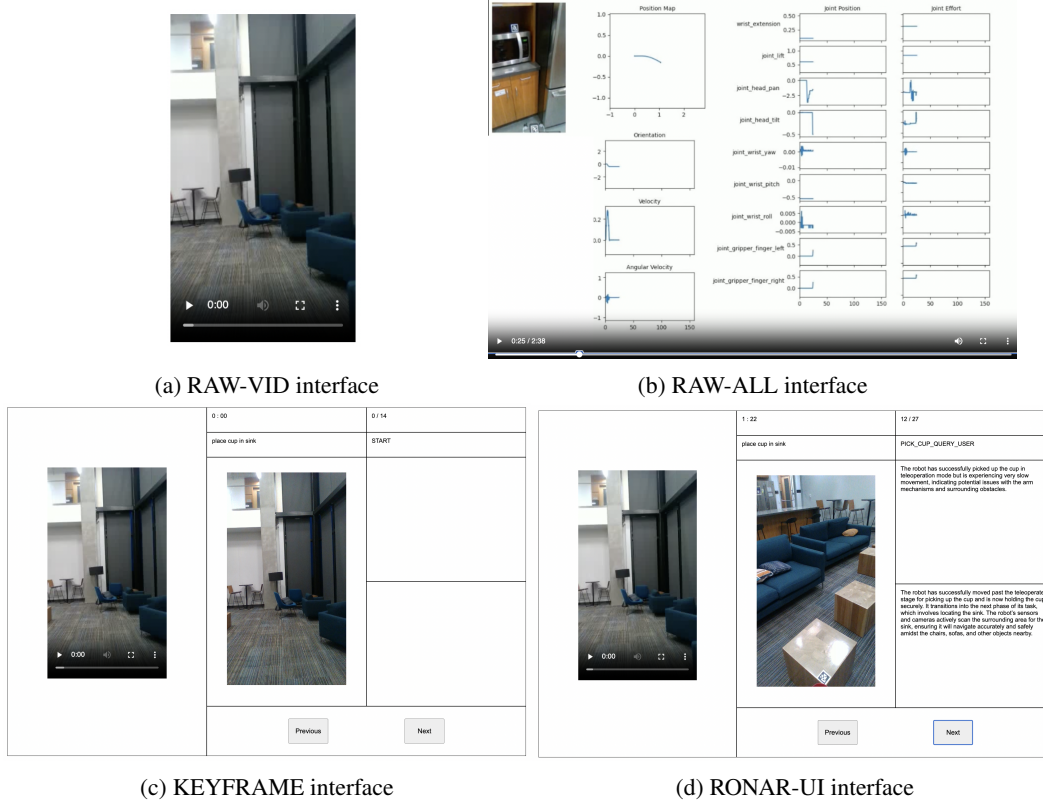


Figure 6: Four interfaces used for failure identification in user study.

LLM, TEM-VLM and RONAR. We pair the three selected frames with the corresponding narrations generated by different methods. In order to prevent biases from the orders of methods, for each participant, we generate a random order of methods and all the names of the methods are hidden. After the participants saw all three image-narration pairs, they can give scores on the four metrics for the corresponding method. Since the order is very critical and users might change their mind by seeing the following narrations, we allow the users to go back and change their ratings by seeing the new narrations.

C.2.3 Failure Identification Using Narration

The second user study is to evaluate the effectiveness and efficiency of narrations on failure identification. The key question we want to answer is how effective and efficient the narration can help users to identify the failures. We design two tasks for the failure identification problem:

- **Failure time identification:** How accurate and efficient can participant correctly identify the time of failure in a demo?
- **Failure explanation:** How accurate and efficient can participant have a reasonable explanation about the failure?

In this study, we design four interfaces which the users will be used to identify the failure time and failure reason. The four failure identifications interfaces are:

- **RAW-VID:** a traditional video player interface which only shows the raw video captured by the robot camera.
- **RAW-ALL:** a video player interface which displays the raw video and all raw sensor readings (both joint and base) from the robot. The sensor readings are visualized by line plots and synchronized with the raw video.

- 247 • **KEYFRAME:** the RONAR-UI interface without narration. It includes the raw video, se-
248 lected keyframes, and state information.
- 249 • **RONAR-UI:** the RONAR-UI interface with fully functionalities. It includes both narra-
250 tions with alert mode and info mode generated RONAR.

251 The appearances of all interfaces can be found in Figure 6. The effectiveness and efficiency are
252 measured by *accuracy* and *time spent* on each of the task. For failure time identification, we have
253 the ground truth failure time and set a time tolerance for each of the demonstrations. If the participant
254 gives an estimated time within the time tolerance, it is marked as correct. Otherwise, it is marked
255 as incorrect. For failure explanation, we ask the participants to write 1-2 sentences to explain the
256 failure in each of the demonstration. Then, three robot experts evaluate the users' answer based
257 on reasonableness independently. The accuracy on failure explanation is the average of rate of
258 correctness marked by the experts. For measuring efficiency, we record the time which participants
259 used to fulfill each task. This time is measured by an expert supervisor with a stopwatch beside the
260 participant. For failure time identification, the start time is when the user click the play button of the
261 video and the finish time is when the user finish typing and click *next* button. For failure explanation,
262 the start time is when the user saw the ground truth failure time and the finish time is when the user
263 finish typing and click *next*.

264 This study is also consist of two parts: an introductory tutorial and a thorough evaluation of inter-
265 faces. In the tutorial, we prepare examples of each interface for users to interact with. We give
266 enough time for them until they are familiar with all the interfaces and comfortable to continue for
267 the actual evaluation. For the formal evaluation, we select four demonstrations from the task, *put*
268 *cup in sink*, with failures in different states and we make sure that each demonstration only contains
269 one failure. These are the failures for the formal evaluation. In order to prevent the biases from
270 the demo-interface pairs, we make a full set of permutations of the four demos and four interfaces,
271 which results 24 permutations. We assign each different permutation to different participants, which
272 covers exact 24 participants.

273 D Experience Summary and Narration Examples

274 D.1 Experience Summary Example

Environment Summary

I observe:

stool_0: 0.93 meters and in front of the robot, left of sofa_0, right of coffee table_0.

sofa_0: 1.13 meters and in front of the robot, right of chair_0, left of chair_1, right of stool_0, left of coffee table_0.

chair_1: 1.24 meters and above the robot, right of chair_0, left of sofa_0, left of stool_0.

chair_0: 1.25 meters and above the robot, left of sofa_0, left of stool_0, right of coffee table_0.

coffee table_0: 1.27 meters and in front of the robot, right of chair_0, right of chair_1, left of sofa_0, right of stool_0.

Internal Summary

Here is what I am doing:

wrist_extension:

Descriptions: position at 19.23% of its maximum extension, velocity nearly at a standstill with a very slight retraction (-0.000226 m/s), effort is very minimal (4.39623047631315e-45%).

Grounded: My arm is mostly retracted and not moving much.

joint_lift:

Descriptions: position at 54.59% of its maximum height, velocity nearly at a standstill with a very slight downward motion (-0.000206 m/s), effort at 40.91% of maximum torque.

Grounded: My arm is raised to about halfway up and holding steady.

joint_head_pan:

Descriptions: position at 76.49% of its maximum left pan, velocity at standstill (0.0 rad/s), effort at 0.0%.

Grounded: My camera is facing mostly straight ahead and not moving.

joint_head_tilt:

Descriptions: position at 64.78% of its maximum upward tilt, velocity at standstill (0.0 rad/s), effort at 0.0%.

Grounded: My camera is tilted slightly upwards and not moving.

joint_wrist_yaw:

Descriptions: position at 15.97% of its maximum yaw to the right, velocity at standstill (0.0 rad/s), effort at 0.0%.

Grounded: My wrist is slightly rotated to the right and not moving.

joint_wrist_pitch:

Descriptions: position at -94.22% of its maximum downward pitch, velocity at 0.0 rad/s, effort at -0.10%.

Grounded: My wrist is pitched downward quite a bit and holding steady.

joint_wrist_roll:

Descriptions: position at 50% of its range, velocity at 0.0 rad/s, effort at 0.0%.

Grounded: My wrist is in a neutral roll position and not moving.

joint_gripper_finger_left:

Descriptions: position at 50% of its opening range, velocity at 0.0 rad/s, effort at 0.0%.

Grounded: The left gripper finger is halfway open and not moving.

joint_gripper_finger_right:

Descriptions: position at 50% of its opening range, velocity at 0.0 rad/s, effort at 0.0%.

Grounded: The right gripper finger is halfway open and not moving.

position:

Descriptions: x at 1.627e-5 m, y at -4.526e-11 m.

Grounded: My base is almost at the same spot, with very minimal movement.

orientation:

Descriptions: orientation at -0.0000289 radians.

Grounded: I am nearly in the same direction I started.

velocity:

Descriptions: velocity moving forward at 0.000166 m/s.

Grounded: I am barely moving forward.

angular_velocity:

Descriptions: angular velocity rotating clockwise at -0.00187 rad/s.

Grounded: I am turning right very slowly.”

Planning Summary

My plan is:

Task: Pick and place the cup in the sink:

Description: The robot should navigate to the table with dirty cup on it, pick up the cup, find the sink, navigate to the sink and place the cup in the sink.

Overall plan:

- Step 1: Begin operation from the START state.
- Step 2: Navigate to the location of the cup.
- Step 3: Execute pick operation to grab the cup.
- Step 4: Navigate to the location of the sink.
- Step 5: Detect the precise position to place the cup.
- Step 6: Place the cup in the sink.
- Step 7: End the operation and transition to END state.

Current state: NAVIGATE_TO_CUP

State History: [['START', 'success']]

D.2 Narration Examples

We have shown narration examples generated by RONAR with different modes in the main paper. In this section, we want to show some interesting cases the narration can help users to identify the status of the robot and failures the robot is experiencing.

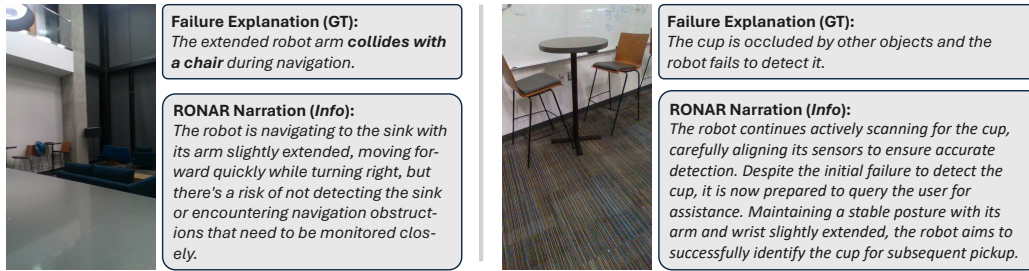


Figure 7: Narration examples generated by RONAR.

278

As shown in Figure 7, the left example shows an interesting failure case which cannot be identified by the vision system alone. In the left example, the robot arm is extended and the arm hits a chair which impedes the movement of the robot. As shown in the frame, users cannot identify what the failures the robot is current facing. During the user study, most participants reason the failure is caused by the robot cannot successfully find and locate the sink, which is due to the detection and mapping errors. The RONAR narration successfully captures the robot arm status ("*its arm slightly extended*") and the movement of the robot ("*moving quickly while turning right*"). As well, it notifies the users of the potential risks which could cause failures. It successfully identifies the risks of navigation obstruction during the task execution without direct visual information.

In the right example, the robot fails to detect the cup with the first attempt. From the visual inputs, it is hard to identify what the robot is doing. Most users consider the robot is navigating to the sink even by watching the video. The narration generated by RONAR successfully captures the past experiences of the robot and explains clearly what the robot is currently doing. In summary, demonstrated by the user studies, it shows significant improvements for users on understanding robot behaviors by using the narrations generated by RONAR.

D.3 Extensions of Narration

The use of narrations uss not only limited to failure analysis and transparency. There could be much more applications and extensions. We study some of the extensions and welcome for more follow-up research on related studies.

D.3.1 Trajectory Summarization

The narration generated by RONAR is event-level, which means each of the narration corresponds to a single key event. It only captures a snapshot of the process, but cannot show an overview of the demonstration. Therefore, we create a higher level summarization, trajectory summarization, which is generated by LLMs using the narration history and captures the details of the trajectory in a human readable way.

It enables the question and answering capability of robot trajectories and create richer ways of interactions with robot data. One potential use of the trajectory summarization is to have customized trajectory retrieval. As shown in Figure 9, for a collection of robot demonstrations, RONAR can generate corresponding trajectory summary for each trajectory. Then, these summaries can be used for trajectory retrieval purposes. Users can search and retrieval trajectories with customized queries. These queried trajectories can be used for model training for various types of learning algorithms, such as imitation learning. It makes robot data search and retrieval much more efficient and accessible.

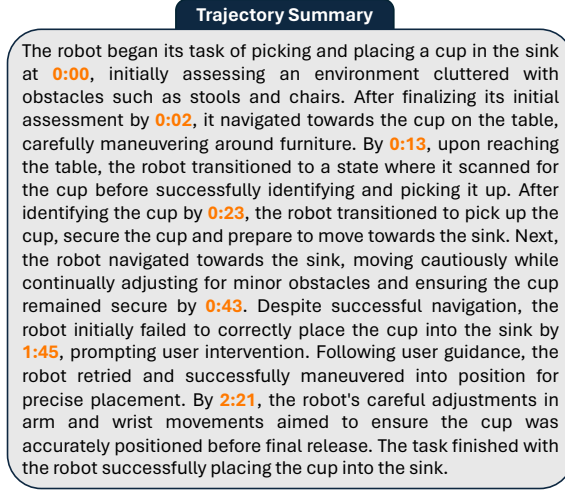


Figure 8: Trajectory summarization

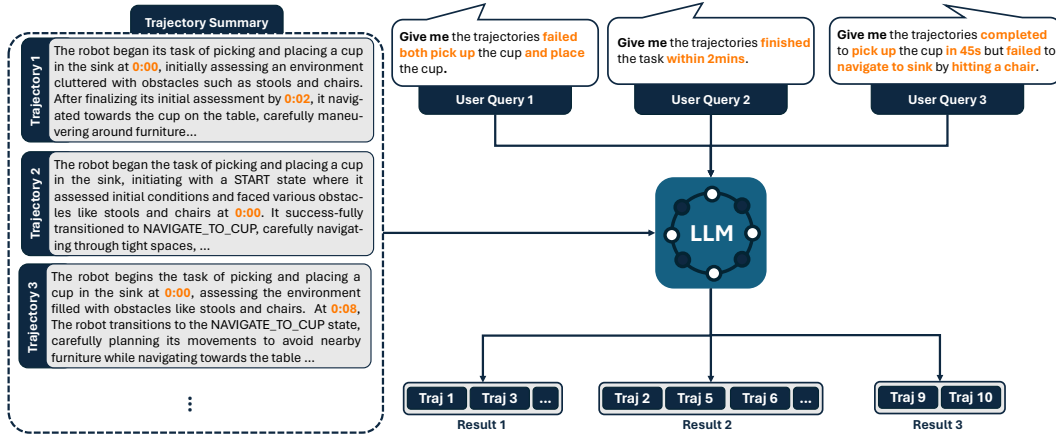


Figure 9: **A pipeline for customized trajectory retrieval.** For each demonstration, RONAR can generate trajectory-level summarization by using event-level narrations. This summarization contains the detailed information of the trajectory and users can retrieve trajectories with customized queries. These customized trajectories can be used for further downstream tasks, such as model training and system analysis.

D.3.2 System Overview

Not limited to trajectory-level summaries, RONAR can generate summaries in an even higher-level. With a collection of trajectory summaries, RONAR can generate a system-level summary to give users an overview of the robot system. As shown in Figure 10, users can generate robot system overview by using a collection of trajectory summaries and RONAR. The system overview is customizable based on users' requirements. In this example, we ask RONAR to generate system overview on failures, recoveries and improvement recommendations based on the experiments. The

329 system overview can give users a big picture of the overall system and assist to make improvements.
330 As well, users can also compare system overviews between different experiment dates to keep track
331 of the system improvement progress.

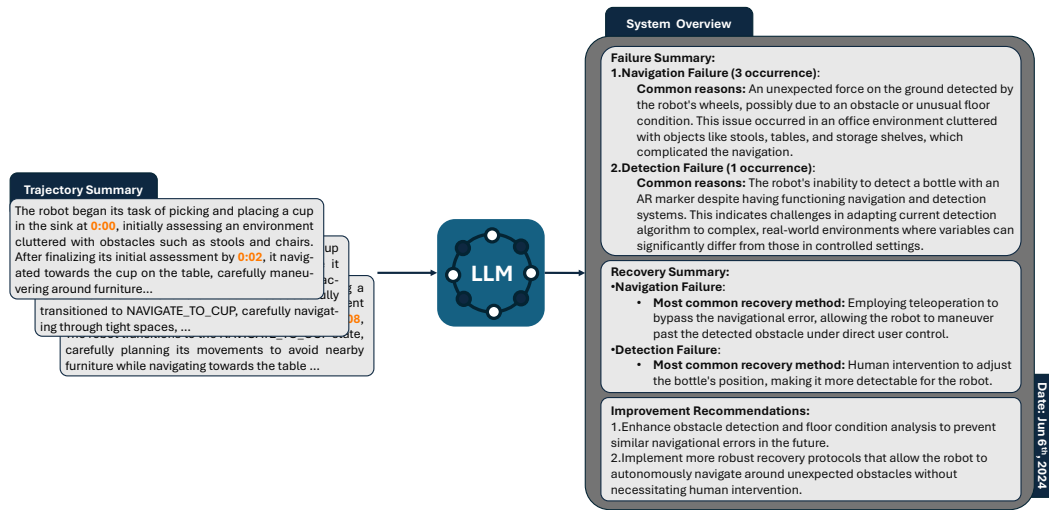


Figure 10: System overview generated by RONAR