

Using Explainable AI and Hierarchical Planning for Outreach with Robots

Rushang Karia*, Jayesh Nagpal*, Daksh Dobhal*,
Pulkit Verma, Rashmeet Nayyar, Naman Shah, Siddharth Srivastava

School of Computing and Augmented Intelligence
Arizona State University

{Rushang.Karia,jnagpal1,ddobhal,verma.pulkit,rmnayyar,npshah4,siddharths}@asu.edu

Abstract

Understanding how robots plan and execute tasks is crucial in today’s world, where they are becoming more prevalent in our daily lives. However, teaching non-experts, such as K-12 students, the complexities of robot planning can be challenging. This work presents an open-source platform, JEDAI.Ed, that simplifies the process using a visual interface that abstracts the details of various planning processes that robots use for performing complex mobile manipulation tasks. Using principles developed in the field of explainable AI, this intuitive platform enables students to use a high-level intuitive instruction set to perform complex tasks, visualize them on an in-built simulator, and to obtain helpful hints and natural language explanations for errors. Finally, JEDAI.Ed, includes an adaptive curriculum generation method that provides students with customized learning ramps. This platform’s efficacy was tested through a user study with university students who had little to no computer science background. Our results show that JEDAI.Ed is highly effective in increasing student engagement, teaching robotics programming, and decreasing the time need to solve tasks as compared to baselines.

1 Motivation

Recent advances in Artificial Intelligence (AI) have enabled the deployment of *programmable* AI robots that can assist humans in a myriad of tasks. However, such advances will have limited utility and scope if users need to have advanced technical knowledge to use them safely and productively. For instance, a mechanical arm robot that can assist humans in assembling different types of components will have limited utility if the operator is unable to understand what it can do, and cannot effectively re-task it to help with new designs.

This paper aims to develop new methods that will allow educators and AI system manufacturers to introduce users to AI systems on the fly, i.e., without requiring advanced degrees in CS/AI as prerequisites. These methods allow for introducing robotics programming to novices.

Our contribution We accomplish our overall objective by introducing JEDAI.Ed, a web application that abstracts the intricacies of robotics programming and exposes the user to an easy-to-use interface to the robot. JEDAI.Ed incorporates several new features that enable its use in educational settings. Firstly, JEDAI.Ed provides an adaptive cur-

riculum design module that can automatically generate problems catered to a particular user by keeping track of the user’s performance. Our system identifies multiple causes of failure and explains them. Finally, JEDAI.Ed utilizes large language models (LLMs) not to discover information but to express factual information and justifications computed using well-defined reasoning processes thereby ensuring the reliability of information being provided.

We implemented JEDAI.Ed by using the existing JEDAI system (Shah et al. 2022) as a baseline. While the core JEDAI system provides a good foundation for development, it has not been developed or evaluated with the components necessary for introductory AI education. E.g., it indirectly requires the users to have some knowledge of robot simulators to operate, does not help educators with designing curricula, etc. Our contributions (mentioned above), along with several other quality-of-life improvements, such as an improved user-interface, etc., make JEDAI.Ed a significant improvement over JEDAI.

We showcase the usefulness of JEDAI.Ed through a user study designed to assess and evaluate its utility and compare it to JEDAI. Our results show that JEDAI.Ed makes robots easy to use and piques curiosity about AI systems. Furthermore, there is a 20% improvement in solution times and significantly higher positive sentiment compared to JEDAI. Furthermore, we have also piloted JEDAI.Ed in two high-school classes and have received positive feedback showcasing the usefulness of JEDAI.Ed across different age groups.

2 Background

In this section, we give a background of key concepts that allow users to program robots for accomplishing tasks.

Running example Consider a robot that is deployed at a coffee shop to help with its day-to-day operations. Depending upon the day’s priorities, the owner may want to program the robot to assist with different tasks such as delivering coffee to customers or washing the cups, etc. To effectively assist the owner, the robot must be able to be “given” tasks (or instructions) by the owner and autonomously perform them.

Planning Robots (and humans) often accomplish tasks by computing a fixed sequence of instructions and then executing them sequentially. These sequences are known as *plans*, *planning* is the process of computing such plans, and algorithms that do planning are called *planners*. Planners take an

*These authors contributed equally.

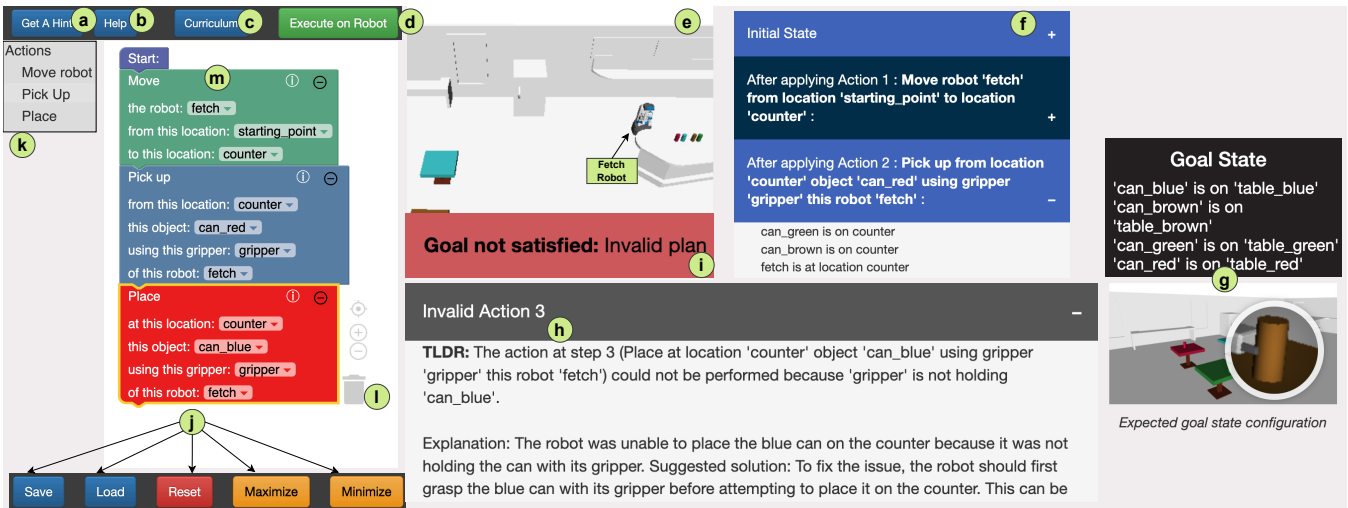


Figure 1: A screenshot (best viewed in color) of the JEDAI.Ed user interface (UI) (zoomed in and enhanced for clarity). The annotated circles describe the different sections of the UI (described in Sec. 3.2). The supplementary material includes unmodified screenshots along with a video walkthrough of the JEDAI.Ed interface.

input task and instruction set and output a plan (consisting of instructions from the instruction set) that solves the task. A *valid* plan for a task is a sequence of semantically consistent instructions starting from the initial state.

Robot instructions and motion planning Robots can only execute a specific type of *low-level* plan known as a *motion plan*. This plan specifies a sequence of movements for each joint of the robot and is obtained using *motion planning*. E.g., the Fetch robot in Fig. 1e has an arm with 8 joints: $\theta_1, \dots, \theta_8$. A motion plan, $\langle [\theta_1^1, \dots, \theta_8^1], \dots, [\theta_1^n, \dots, \theta_8^n] \rangle$, that uses the robot to accomplish an example task of picking up a coffee cup from the counter would contain a sequence of *low-level* instructions $[\theta_1^i, \dots, \theta_k^i]$ that contain numeric values, $\theta_k \in \mathbb{R}$, for all of its joints. Computing such low-level instructions needs robot-specific knowledge and requires complex algebraic arithmetic to compute a motion plan that provides smooth (and safe) motion. These constraints make motion planning quite difficult for humans.

Human instructions and plans Contrary to robots, humans typically accomplish tasks by following instructions at a higher level of abstraction than robots. E.g., to accomplish the same task described in the preceding paragraph, a human often computes a *high-level* plan, $\langle \text{Go to the counter, Pick up the coffee cup} \rangle$, consisting of *high-level* instructions. Humans can find (and execute) high-level plans for complex tasks fairly easily, however, robots can not use such plans directly to accomplish tasks.

Hierarchical planning Given the difficulty of motion planning, it is easy to see that programmable robots must accept high-level instructions to be usable by humans. In this work, we focus on human-in-the-loop (HITL) robot programming where high-level plans are provided by a human and a hierarchical planner converts such plans into a sequence of motion plans that the robot can execute.

Explaining Failures This tiered approach to HITL robotics programming introduces some new hurdles. One

key challenge is that high-level plans might not be successfully compiled into low-level plans. E.g., a high-level plan $\langle \text{Pick up the coffee cup} \rangle$ cannot be compiled into a low-level plan for a single-arm robot if it is already holding something else. When such failures occur, it is imperative that the robot appropriately informs the user of the failure in high-level terms that the user can easily understand. Explaining why a failure occurred can allow a user to correct (or modify) the high-level instructions so that the desired behavior can be achieved. E.g., an explanation of the form “*I (the robot) cannot pick up the coffee cup because I am currently holding a water bottle*” allows the user to (a) identify why the robot could not accomplish the task, and (b) modify their instructions so that the robot can accomplish it.

3 The JEDAI.Ed Platform

We aim to develop a platform that makes robotics programming accessible to a wide spectrum of users and use cases. Thus, we have taken several design considerations (detailed in the supplement) to develop JEDAI.Ed, an open source¹ pedagogical tool that brings robotics programming into the hands of novice users. JEDAI.Ed is usable by educators seeking to teach classes on AI, by hobbyists who are interested in robotics, and many others. We compare JEDAI.Ed with JEDAI w.r.t. some of the desiderata in Table 1.

The next section discusses JEDAI.Ed’s features that make it an ideal pedagogical platform for robotics programming followed by an example use case of JEDAI.Ed for programming a robot on a task from our user study.

3.1 Learning Objectives

The objective for JEDAI.Ed is to facilitate the understanding of reasoning and quickly provide high-level instructions to robots to perform tasks. Furthermore, our platform explains

¹We plan to publicly release the source code post acceptance.

Desiderata	JEDAI.Ed	JEDAI
Open source	✓	✓
Minimal system requirements	✓	✓
Integrated simulation	✓	✓
Intuitive user interface	✓	✗
Adaptive problem generation	✓	✗
Multi-failure explanations	✓	✗
LLM-powered NL explanations	✓	✗

Table 1: A comparison of some of the features of JEDAI.Ed compared to JEDAI. A detailed description of the desiderata is available in the supplementary material.

failures and thus allows users to learn more about the capabilities of the robot. Our focus ties well with the objective of the AI4K12 Big Idea 2 – Representation & Reasoning² which requires users to be able to reason about how their instructions can change the state of the world and use this knowledge to compute to plan.

3.2 System Overview

The JEDAI.Ed architecture, illustrated in Fig. 2, is modular in design allowing for easy customization (discussed in supplement). Fig. 1 shows the overall JEDAI.Ed interface that is presented to users. The JEDAI.Ed user interface module (UI) is the front-end that users interact with and can be run on any modern web-browser making it widely accessible. The back-end can be hosted on any server.

User Interface (UI) The JEDAI.Ed UI follows the single-page application (SPA) design methodology providing the user with all pertinent information on a single page thereby reducing navigation fatigue. Users are presented with a playground area where they can utilize the intuitive, high-level instruction sets (Fig. 1*k*) to create plans via Blockly (Google 2018) – a block-based programming language. For example, the *Move* action in Fig. 1*m* represents the high-level instruction ‘*Move the robot fetch from the starting point to the counter*’. Finally, JEDAI.Ed provides feedback via different modalities (e.g., an audio click when blocks are connected, changing the color of invalid blocks to red, etc.).

Low-Level Module (MPM) JEDAI.Ed provides an integrated low-level planner, ATAM (Shah et al. 2020) and simulator, OpenRAVE (Diankov 2010) for executing user-provided plans on a robot using the UI (Fig. 1*d*). ATAM converts high-level plans to low-level plans that can be executed and visualized on the UI via the simulator (Fig. 1*e*). This execution is a close approximation of the real-world. The motion planning process is streamed in real-time providing informative insights about it. We include one such execution in the video walkthrough included in the supplement.

User Assistance Module (UAM) It is well-known that iteration and improvement are part of the learning process and learning from failures can be expected in an educational setting (Jackson et al. 2022). JEDAI.Ed uses advances in ex-

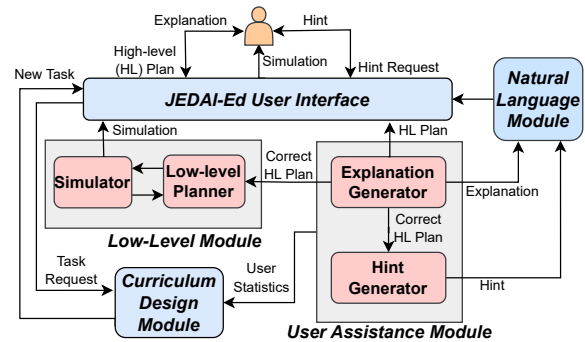


Figure 2: The JEDAI.Ed architecture (described in Sec. 3.2).

plainable AI to automatically generate explanations that allow users to (a) learn why their plans are failing, (b) better understand the robot’s limitations and capabilities, and (c) fix their plans so that the robot can accomplish the tasks.

Explanations: Our system uses HELM (Sreedharan, Srivastava, and Kambhampati 2018) and VAL (Howey, Long, and Fox 2004) for generating explanations whenever failures occur in the user-submitted high-level plan. Once a user connects any block, the current plan is routed through these components to identify whether the plan is valid. An invalid plan is passed to HELM and VAL to generate formal explanations which are then translated to NL via templates and LLMs and displayed to the user.

Natural language module (NLM) This module processes messages from all components, converting them to a human-readable message via hand-coded NL templates and/or LLMs. We use well-defined reasoning processes to avoid hallucinations in LLMs by utilizing LLMs primarily as translators and not as reasoners. In our experiments, we used GPT-3.5 Turbo (OpenAI 2022). However, any LLM can be easily configured. A detailed description of our prompts and NL templates is available in the supplementary material.

Ethical Considerations and Guardrails: We limit the potential of LLMs in generating offensive content by (a) using fixed prompts and not free-form chat mode, and (b) using LLMs that are compliant with the OpenAI content policy which dictates the types of responses the LLM can generate. JEDAI.Ed does not include any user-identifiable information in requests to the LLMs nor is such information required for using any feature of JEDAI.Ed.

Curriculum Design Module (CDM) JEDAI.Ed includes several environments such as Cafeworld, Towers of Hanoi, etc. that are widely used in AI coursework (Fig. 3). These environments provide a diverse mix of tasks and robots that instructors can use as activities for teaching AI planning. Furthermore, JEDAI.Ed also includes a problem generator that can generate new problems on-the-fly by utilizing breadth-first search (BFS) (Russell and Norvig 2020) and allow users to explore the capabilities of the robot on their own.

Adaptive Problem Generation: Our platform develops an adaptive problem generation module to help novices understand the capabilities of the robot in a systematic and directed fashion. We do so by keeping track of the user’s per-

²<https://ai4k12.org/big-idea-2-overview/>

formance as they solve problems and generating new problems (in the same environment) that focus on aspects that the user has had difficulty with. For example, actions that the user has made mistakes on. The next problem focuses on generating problems that only require the user to use the difficult action thereby reducing the overall cognitive load and making learning easier (Moos and Pitton 2014). Our overall process for doing so is indicated in Alg. 1. Intuitively, we increase the cost of actions that the user performs well at and decrease the cost of actions that the user has difficulty with. We then use BFS to generate a new problem such that at least one difficult action is covered. The random problem generator described earlier also performs BFS but assumes all actions have equal costs whereas with adaptive BFS, action costs are different and consequently problems generated are not random but directed. This method also works well in a coldstart setting since it initially assumes that the user is not proficient at any action (i.e., $\forall a C_u[a] = 0$). Additional details of this process are included in the supplement.

Walkthrough: Using JEDAI.Ed for a programmable single-arm, mobile robot We now describe a typical session of JEDAI.Ed that introduces the functionalities of a mobile manipulator like Fetch (Fig. 1e) that is intended to be used in a coffee shop based on the running example (Sec. 2). We also used this in our user-study and the walkthrough describes the typical processes involved.

First, the educator installs the JEDAI.Ed system on a machine. Next, the educator uses the CDM to select an appropriate environment for the students (e.g., Coffee Shop). The educator then generates (or selects preset) tasks for the student to accomplish (CDM). Alternatively, the educator could instruct the students to use the adaptive problem generator and then solve a test task. The student accesses JEDAI.Ed on a web browser and begins learning.

The UI presents the user with the necessary information such as the task description and goal (Fig. 1g), available instruction set (Fig. 1k), and a simulator window (Fig. 1e). The goal is provided both in textual as well as visual descriptions along with a magnifier to view finer details of the image. A *Help* button (Fig. 1b) provides useful descriptions about the interface and is available to the user at all times.

The user then uses the instruction sets along with intuitive knowledge to create a plan of high-level instructions by dragging-and-dropping Blockly blocks (Fig. 1l) and connecting them to the *Start* block. An audible click lets the user know that the block snapped to another block.

Every connected block is checked for validity in real-time and explanations (UAM) are provided if the user’s current plan contains any invalid actions (Fig. 1h). E.g., the explanation shown in Fig. 1h explains that the instruction ‘Place the blue can at the counter using gripper of the Fetch’ failed because the robot was not holding the blue can. The user may also check the result of their current plan in the state display area (Fig. 1f). The user may also request a hint (UAM, Fig. 1a, elaborated in the supplement) that returns a high-level instruction as a pop-up message.

Once a valid high-level plan (irrespective of whether it accomplishes the goal or not) is achieved (Fig. 1g), the “Ex-

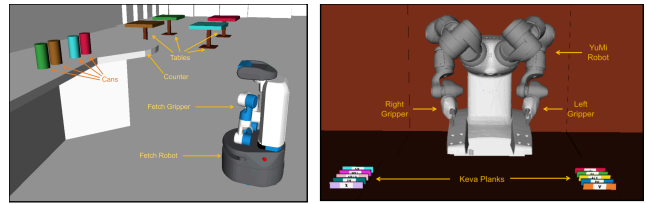


Figure 3: Example environments, Coffee Shop (left) and Keva (right), included with JEDAI.Ed. These environments feature sample tasks and problem generators for use as activities. More environments and their details are included in the supplement.

Algorithm 1: Adaptive User-Performance Tracking

Input: user-performance map C_u , action a , was-hinted h
Output: Updated user-performance map C_u

```

1  $s \leftarrow \text{getCurrentState}()$ 
2 if  $\text{canExecuteAction}(s, a)$  and not  $h$  then
   | // User knows action: Cost  $\uparrow$ 
3   |  $C_u[a] \leftarrow C_u[a] + 1$ 
4 else
   | // User does not know action: Cost  $\downarrow$ 
5   |  $C_u[a] \leftarrow C_u[a] - 1$ 

```

ecute on Robot” (Fig. 1d) button is activated and the user may submit their plan to be executed on the robot. The planning process and real-time execution of the low-level plan are streamed by the simulator (MPM, Fig. 1c).

4 Empirical Evaluation

We developed JEDAI.Ed to expose novice users to AI and robotics. We conducted a user study to evaluate if JEDAI.Ed achieves the goal by evaluating the following hypotheses:

H1 (Increased curiosity): JEDAI.Ed increases the curiosity of users to learn more about robotics and AI.

H2 (Easier programming): JEDAI.Ed makes it easy for users to provide instructions to robots.

H3 (Improved understanding): JEDAI.Ed improves user understanding w.r.t. the limitations/capabilities of a robot.

H4 (Helpful explanations): JEDAI.Ed’s provided explanations help users understand (and fix) errors in their plans.

H5 (Intuitive UI): JEDAI.Ed’s UI is intuitive and easy to use requiring little to no study facilitator intervention.

H6 (Programming confidence): JEDAI.Ed increases users confidence in instructing robots to accomplish tasks.

H7 (Faster solving): JEDAI.Ed allows users to solve tasks faster than JEDAI.

To evaluate the validity of these hypotheses, we designed a user study for evaluating JEDAI.Ed and comparing it with JEDAI. We present the study methodology below.

4.1 User Study Setup

We hired 43 university students with no background in computer science as participants for an IRB-approved user study. We discarded 1 incomplete/invalid response, resulting in a

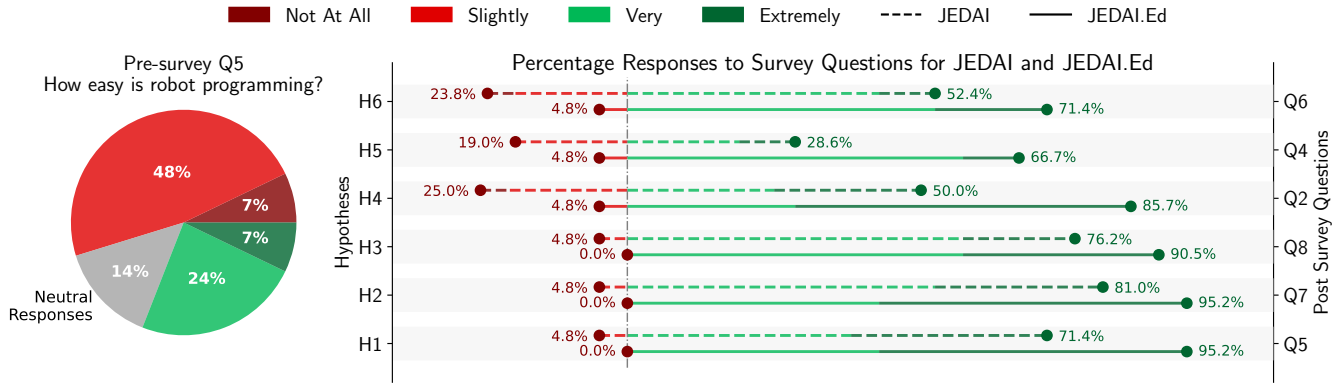


Figure 4: Results from our user study with two control groups ($n = 42$) split evenly between JEDAI.Ed and JEDAI. The x-axis plots the responses as a percentage. Green (red) bars to the right (left) indicate positive (negative) sentiment. Stem annotations indicate the total positive (negative) sentiment. The left y-axis specifies the target hypothesis. The right y-axis specifies the question that assesses its validity.

Hypothesis	Post Survey Question used for Testing Hypothesis	$\mu \pm \sigma$ w.r.t. 1-sample t-test ($\mu_0^1 = 2$)	
		JEDAI.Ed	JEDAI
H1: Increased curiosity	Q5: As compared to before participating, how much has your curiosity increased to learn more about AI systems and robots?	3.47 ± 0.60	3.00 ± 0.89
H2: Easier programming	Q7: Do you agree that the JEDAI.Ed system made it easier for you to provide instructions to a robot for performing tasks?	3.47 ± 0.60	3.04 ± 0.80
H3: Improved understanding	Q8: Do you agree that JEDAI.Ed helps improve the understanding of the robot’s limitations and capabilities?	3.23 ± 0.62	2.90 ± 0.76
H4: Helpful explanations	Q2: How helpful were the explanations that were given for the cause of an error?	3.38 ± 0.86	2.42 ± 1.20
H5: Intuitive UI	Q4: How intuitive was the interface?	2.71 ± 0.71	2.19 ± 0.87
H6: Programming confidence	Q6: How well do you think you now understand how one can use an AI system to make a plan for a robot to perform a task?	2.85 ± 0.79	2.33 ± 1.06

Table 2: JEDAI.Ed user study results ($n = 42$) used to validate our hypotheses. The table provides a short description of the target hypothesis, the corresponding questions used to validate it, and the one-sample t-test results. All results are statistically significant ($p < 0.05$) except for entries in **bold; red**. Comprehensive statistical data is available in the supplement.

sample size of 42. 23 of these were from a non-STEM background. We divided the participants into two control groups. The first (second) control group was assigned the JEDAI.Ed (JEDAI) system for use in the study. The study lasted 45 minutes, was conducted in-person, and had four phases:

Pre-survey phase (8 min): Participants were presented with an introductory video about AI. Next, to acquire a detailed understanding of the participant’s background, interests in AI, level of awareness and engagement with AI technologies, we employed a pre-survey questionnaire.

Training phase (12 min): This phase was intended to get users familiarized with the system and tasks. Communication with the study facilitator was allowed. Participants were presented sequentially with three tasks of the *Coffee shop* environment (Sec. 2) each of which involved utilizing a Fetch robot to deliver cans to tables. JEDAI.Ed used the adaptive problem generation algorithm to generate training tasks. We used randomly generated training tasks for JEDAI.

We ensured that all generated training tasks needed 50% fewer instructions to accomplish than the test task.

Test phase (12 min): The participants solved a test task during this phase. The test task was much harder than the training tasks and required users to deliver multiple cans (optimally using 16 high-level instructions). No communication with the study facilitator was allowed during this phase. The participants were then asked to complete a post-survey questionnaire whose questions were designed to obtain the participant’s opinion on the platform they interacted with and to determine if their interest and curiosity had increased post-use. We also collected system logs for analytics data.

Sentiment change phase (13 min): This phase is intended to analyze the sentiment change after interacting with both JEDAI.Ed and JEDAI. In this phase, participants who interacted with JEDAI.Ed (JEDAI) in the previous phases were asked to interact freely with JEDAI (JEDAI.Ed). They were once again asked to answer a post-survey questionnaire.

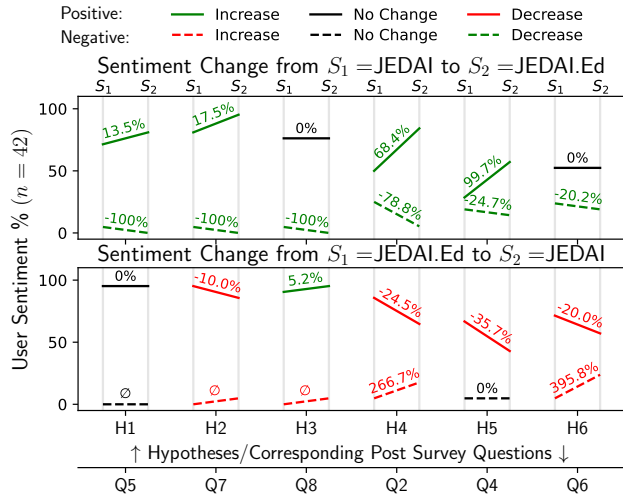


Figure 5: Slope charts for results from our sentiment change phase. Users interacted with S_1 first and S_2 next. The y-axis shows the absolute user sentiment while annotations on the line plots show the % improvement ($\frac{S_2 - S_1}{S_1} \times 100$). We use \emptyset when $S_1 = 0$.

This questionnaire was the same as that of the test phase but they could not see the previous responses.

Questionnaire methodology All responses to the questions used the Likert Scale (Likert 1932) to provide a more intricate depiction compared to binary responses.

Hypothesis testing Our Likert Scale data was converted to values from 0 to 4 with 0 (4) being the most negative (positive) response and 2 being neutral. We used the p-value obtained by using the one-sample t-test (Ross and Willson 2017) to test the statistical significance. Within a control group, we assumed the data to be two-tailed for the questions used to validate the hypotheses and used the hypothesis of no difference, i.e., $\mu_0^1 = 2$ (balanced Likert scales), as the null hypothesis. Thus, for statistically significant results $\mu_0^1 > 2$ denotes positive sentiment and vice versa.

4.2 Study Results

Fig. 4 and Table. 2 show our results, the survey questions used to analyze the hypotheses, and data from the statistical tests. All data was statistically significant for JEDAI.Ed. Moreover, JEDAI.Ed’s $\mu \geq 2.7$ showcases an improved, positive experience. We analyze our results below.

H1 (Increasing curiosity): Fig. 4 shows that after interacting with JEDAI.Ed, user curiosity is 95% positive. This is far greater than JEDAI, whose positive user sentiment is 71%.

H2, H3, and H6: Our pre-survey results (Fig. 4, pie) show that before using JEDAI.Ed, 55% of users believed that robot programming was not easy.

H2 (Easier Programming): 95% of users thought that JEDAI.Ed made it easier to program robots. In contrast, JEDAI only managed to increase positive sentiment to 71%.

H3 (Improved understanding), H6 (Programming confidence): After interacting with JEDAI.Ed, 90% of users think that they better understand the robot’s capabilities, and 71%

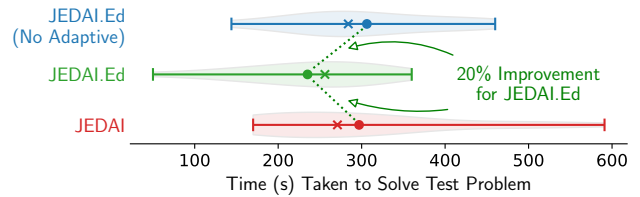


Figure 6: Violin plots that indicate the time needed to solve the test task. \bullet (\times) represents the mean (median).

of users were confident that they could program robots.

H4 (Helpful explanations): Users were extremely positive in their feedback w.r.t. JEDAI.Ed provided explanations ($\approx 86\%$ of users thought that the explanations were helpful). As compared to JEDAI, JEDAI.Ed provides both brief and LLM-based descriptive explanations that better explain why a failure occurred. Thus, JEDAI explanations were rated significantly lower and also had a 25% negative sentiment.

H5 (Intuitive interface): Most users using JEDAI.Ed were able to navigate the interface without any help. JEDAI.Ed is modern and includes many quality-of-life features such as the ability to minimize blocks, etc. which are lacking in JEDAI. Fig. 4 shows that 66% of users found JEDAI.Ed’s UI intuitive as compared to JEDAI which had only 28% positive sentiment and had a 19% negative sentiment.

H7 (Faster solving): Fig. 6 shows the distribution of times required to solve the test task. JEDAI.Ed users were able to solve the test task in 235 seconds which is 20% faster than JEDAI. There were 4 (3) users for JEDAI.Ed (JEDAI) that were not able to solve the test task. One additional JEDAI.Ed user encountered an internal error requiring a system restart thus resulting in them not being counted.

One of the key advantages of JEDAI.Ed is the adaptive problem generation that appropriately adjusts the difficulty of tasks so that users can learn faster. JEDAI.Ed also informs users of multiple invalid actions (and explaining two of them) in real-time as compared to JEDAI where users need to submit plans to get any feedback and are only informed and explained of a single invalid action.

Ablation Study To further investigate the impact of adaptive problem generation, we conducted an ablation study by recruiting an additional 19 students with similar backgrounds. These students were administered the same study using JEDAI.Ed. The only change we made was to use randomly generated problems in the training phase instead of the adaptive problem-generation method employed earlier. Three users in this new study were unable to solve the test task. Our results in Fig. 6 show that without the adaptive problem generation, the training tasks are much harder for the students and consequently they cannot perform as well on the test task. We attribute the similarities between the solve times w.r.t. JEDAI to the fact that JEDAI also explains failures and thus provides similar feedback.

Improved Sentiment over JEDAI Fig. 5 shows that users have a positive (negative) sentiment change across all metrics when interacting with JEDAI (JEDAI.Ed) first and then

experiencing JEDAI.Ed (JEDAI). These observations, along with the rest of our analysis, shows that JEDAI.Ed offers several significant improvements over JEDAI resulting in an overall enhanced user-experience when using JEDAI.Ed as a platform for robotics programming.

4.3 Pilot Program on High School Students

We also demonstrated JEDAI.Ed across 3 sessions at two different high schools to a total of ≈ 90 students. Each session lasted 60 minutes and students were asked to complete tasks across 3 different environments (Coffee shop, Keva π -Planks, and Towers of Hanoi). Most students were able to complete all tasks without any supervision. Pictures from our visits are included in Fig. 7. We also solicited feedback from the program coordinator who mentioned:

“They found it very user-friendly. Thank you again for the visit and looking forward to seeing more in the future.”

4.4 Improvement Opportunities

We now discuss what didn’t work, and improvement opportunities based on feedback from the user study and our pilot.

For our pilot program, we hosted our system on an AWS cloud instance to serve the students. The network latency between the school and the server was visible in the interface and caused latency issues where the video stream of the robot executing the plan was not rendering correctly. Optimizing the motion planner to break down the trajectory packets and send them piece-by-piece would provide for a smoother experience which we are currently implementing.

Some users from had difficulty understanding that plans begin at the *Start* block. They mentioned that renaming it to *Connect blocks here* would improve the UI’s intuitiveness.

We observed that in its current iteration JEDAI.Ed is not widely accessible on devices which do not employ a keyboard and mouse. We plan to improve the accessibility of our platform by using generative AI so that users can provide plans verbally using multimodal models.

An additional feature that we are working on allows users to use programming constructs like loops and conditionals to form their plans. These allow for the inclusion of richer environments with non-deterministic action semantics. Explaining failures in such programs is a challenging and exciting question for future research.

5 Related Work

This work brings together several independent research directions in a single platform. We discuss them here.

Visualizations in planning There are tools that help visualize the planning process to make it is easy to understand for the users. Such tools include Web Planner (Magnaguagno et al. 2017), Planimation (Chen et al. 2019), PDSim (De Pellegrin and Petrick 2021), vPlanSim (Roberts et al. 2021), PlanVis (Cantareira, Canal, and Borgo 2022) etc. These methods focus on visualizing the planning process for users, whereas JEDAI.Ed, in addition, also helps novices in planning on their own, executing the plans on robots, and explaining their mistakes to them.

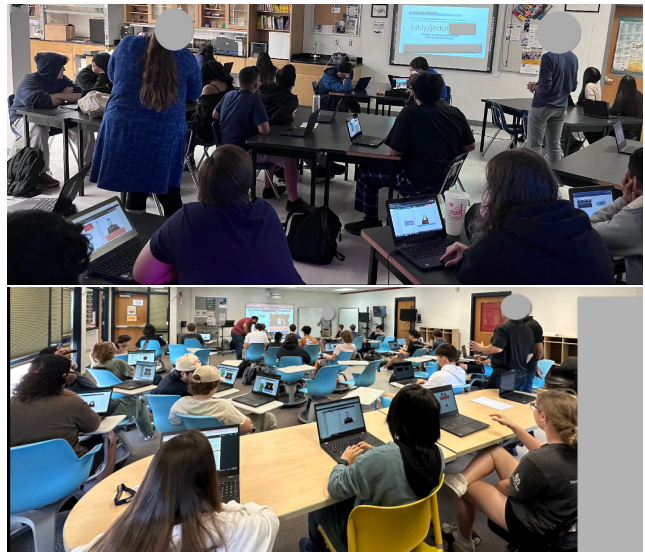


Figure 7: Engagement from JEDAI.Ed’s pilot program on two high schools.

Robot programming interfaces CoBlox (Weintrop et al. 2018) used a similar interface for creating low-level plans for robots but also requires users to provide low-level plans. Winterer et al. (2020) analyzed the use of Blockly for programming industrial robots. These approaches target expert users and unlike JEDAI.Ed cannot be used by novices.

AI concepts for students Robot-VPE (Krishnamoorthy and Kapila 2016) and Code3 (Huang and Cakmak 2017) used a Blockly-like interface for K12 students to write programs for robots. Broll and Grover (2023) created a tool to teach complex ML concepts to students using block-based pre-programmed games. Maestro (Geleta et al. 2023) used goal-based scenarios to teach students about robust AI.

Generating explanations with easy-to-understand interfaces There is a large body of work on generating explanations for user-provided plans. Few such approaches (Grover et al. 2020; Valmeekam et al. 2022; Brandao et al. 2021; Kumar et al. 2022) use an easy-to-understand user interface and natural language to make the explanations easily accessible to novice users. These approaches do not integrate low-level planning and thus cannot be used to program robots.

6 Conclusion

We introduced JEDAI.Ed, an open-source platform to introduce high-level robot planning to novices. We showed that JEDAI.Ed is an effective and intuitive platform in teaching AI planning to users without a background in the subject. JEDAI.Ed significantly improves upon its predecessor and adds several new and novel features. Adapting curriculums tailored to individual users allows for more effective learning which is evident from faster solution times on our platform. Our results show that users prefer JEDAI.Ed over JEDAI. Moreover, JEDAI.Ed was able to successfully engage students and pique their curiosity in learning more about AI planning. Our pilot program was highly successful

and increased the students' confidence in robotics programming. We hope to keep developing and making JEDAI.Ed available to wider audiences.

Acknowledgements

This work was supported in part by the ONR under grant N000142312416.

Ethical Statement

This work involved recruiting humans for our study. Both our pre and post survey questionnaires went through an IRB review process and were approved before starting the study. We ensured that students had little to no background in computer science and only allowed participants who either (a) were not enrolled in a computer science major, (b) did not have any significant programming experience, and (c) had not formally or informally enrolled in a data structures or equivalent course either through a university or an online education platform. For computer science majors, data structures is a pre-requisite for a majority of programming and robotics related classes. Thus, our computer science majors were composed mainly of students in their freshmen year with little to no exposure to any computer science concepts.

Usage of LLMs carries the risk of providing content that might not be relevant or might be offensive to its users. We mitigated this by using OpenAI's latest GPT-3.5-turbo model (gpt-3.5-turbo-0125) which is compliant with the OpenAI usage policy (OpenAI 2024) on content generation. LLMs are more prone to generate irrelevant or offensive content when engaged in a dialogue with users. In our case, our prompts are structured and fixed and thus are unlikely to generate irrelevant text. Additionally, in accordance with the company policy, GPT-3.5 has default content filters that stop any offensive or inappropriate text from being generated and returned to be displayed in JEDAI.Ed.

Finally, with regards to user privacy, no user identifying information was provided to GPT-3.5 or used at any point in JEDAI.Ed and in our experiments with JEDAI.

References

- Brandao, M.; Canal, G.; Krivić, S.; and Magazzeni, D. 2021. Towards Providing Explanations for Robot Motion Planning. In *Proc. ICRA*.
- Broll, B.; and Grover, S. 2023. Beyond Black-Boxes: Teaching Complex Machine Learning Ideas through Scaffolded Interactive Activities. In *Proc. EAAI Symposium*.
- Bylander, T. 1994. The Computational Complexity of Propositional STRIPS Planning. *AIJ*, 69(1-2): 165–204.
- Cantareira, G. D.; Canal, G.; and Borgo, R. 2022. Actor-Focused Interactive Visualization for AI Planning. In *Proc. ICAPS*.
- Chen, G.; Ding, Y.; Edwards, H.; Chau, C. H.; Hou, S.; Johnson, G.; Sharukh Syed, M.; Tang, H.; Wu, Y.; Yan, Y.; Gil, T.; and Nir, L. 2019. Planimation. In *ICAPS 2019 System Demonstrations*.
- De Pellegrin, E.; and Petrick, R. P. A. 2021. PDSim: Simulating Classical Planning Domains with the Unity Game Engine. In *ICAPS 2021 System Demonstrations*.
- Diankov, R. 2010. *Automated Construction of Robotic Manipulation Programs*. Ph.D. thesis, Carnegie Mellon University.
- Geleta, M.; Xu, J.; Loya, M.; Wang, J.; Singh, S.; Li, Z.; and Gago-Masague, S. 2023. Maestro: A Gamified Platform for Teaching AI Robustness. In *Proc. EAAI Symposium*.
- Google. 2018. Blockly. <https://github.com/google/blockly>.
- Grover, S.; Sengupta, S.; Chakraborti, T.; Mishra, A. P.; and Kambhampati, S. 2020. RADAR: Automated Task Planning for Proactive Decision Support. *Human-Computer Interaction*, 35(5-6): 387–412.
- Hoffmann, J. 2001. FF: The Fast-Forward Planning System. *AI Magazine*, 22(3): 57.
- Howey, R.; Long, D.; and Fox, M. 2004. VAL: Automatic Plan Validation, Continuous Effects and Mixed Initiative Planning Using PDDL. In *Proc. ICTAI*.
- Huang, J.; and Cakmak, M. 2017. Code3: A System for End-to-End Programming of Mobile Manipulator Robots for Novices and Experts. In *Proc. HRI*.
- Jackson, A.; Godwin, A.; Bartholomew, S.; and Mentzer, N. 2022. Learning from failure: A systematized review. *International Journal of Technology and Design Education*, 32(3): 1853–1873.
- Krishnamoorthy, S. P.; and Kapila, V. 2016. Using A Visual Programming Environment and Custom Robots to Learn C Programming and K-12 STEM Concepts. In *Proceedings of the 6th Conf. on Creativity and Fabrication in Education*.
- Kumar, A.; Vasileiou, S. L.; Bancelhon, M.; Ottley, A.; and Yeoh, W. 2022. VizXP: A Visualization Framework for Conveying Explanations to Users in Model Reconciliation Problems. In *Proc. ICAPS*.
- Likert, R. 1932. A Technique for the Measurement of Attitudes. *Archives of Psychology*, 22(140): 5–55.
- Magnaguagno, M. C.; Fraga Pereira, R.; Móre, M. D.; and Meneguzzi, F. R. 2017. WEB PLANNER: A Tool to Develop Classical Planning Domains and Visualize Heuristic State-Space Search. In *ICAPS 2017 Workshop on User Interfaces and Scheduling and Planning*.
- Moos, D. C.; and Pitton, D. 2014. Student teacher challenges: using the cognitive load theory as an explanatory lens. *Teaching Education*, 25(2): 127–141.
- noVNC. 2024. noVNC. <https://novnc.com/info.html>. [Online; accessed 2-Feb-2024].
- OpenAI. 2022. GPT-3.5. <https://platform.openai.com/docs/model-index-for-researchers>. [Online; accessed 5-Sept-2023].
- OpenAI. 2024. Usage Policies. <https://openai.com/policies/usage-policies>. [Online; accessed 2-Feb-2024].
- Pearson, E. S.; D'Agostino, R. B.; and Bowman, K. O. 1977. Tests for departure from normality: Comparison of powers. *Biometrika*, 64: 231–246.

- Roberts, J. O.; Mastorakis, G.; Lazaruk, B.; Franco, S.; Stokes, A. A.; and Bernardini, S. 2021. vPlanSim: An Open Source Graphical Interface for the Visualisation and Simulation of AI Systems. In *Proc. ICAPS*.
- Ross, A.; and Willson, V. L. 2017. *One-Sample T-Test*, 9–12. Rotterdam: SensePublishers. ISBN 978-94-6351-086-8.
- Russell, S.; and Norvig, P. 2020. *Artificial Intelligence: A Modern Approach (4th Edition)*. Pearson.
- Shah, N.; Kala Vasudevan, D.; Kumar, K.; Kamojjhala, P.; and Srivastava, S. 2020. Anytime Integrated Task and Motion Policies for Stochastic Environments. In *Proc. ICRA*.
- Shah, N.; Verma, P.; Angle, T.; and Srivastava, S. 2022. JEDAI: A System for Skill-Aligned Explainable Robot Planning. In *Proc. AAMAS*.
- Sreedharan, S.; Srivastava, S.; and Kambhampati, S. 2018. Hierarchical Expertise Level Modeling for User-Specific Contrastive Explanations. In *Proc. IJCAI*.
- Valmeekam, K.; Sreedharan, S.; Sengupta, S.; and Kambhampati, S. 2022. RADAR-X: An Interactive Mixed Initiative Planning Interface Pairing Contrastive Explanations and Revised Plan Suggestions. In *Proc. ICAPS*.
- Weintrop, D.; Afzal, A.; Salac, J.; Francis, P.; Li, B.; Shepherd, D. C.; and Franklin, D. 2018. Evaluating CoBlox: A Comparative Study of Robotics Programming Environments for Adult Novices. In *Proc. CHI*.
- Wilcoxon, F. 1945. Individual Comparisons by Ranking Methods. *Biometrics Bulletin*, 1(6): 80–83.
- Winterer, M.; Salomon, C.; Köberle, J.; Ramler, R.; and Schittengruber, M. 2020. An Expert Review on the Applicability of Blockly for Industrial Robot Programming. In *Proc. ETFA*.

The supplementary material begins with an in-depth treatment of the desiderata mentioned in the main paper. Also included, is a video walkthrough of some of the features of the interface. This is followed by an exposition of the design considerations and extensibility of JEDAI.Ed. We then briefly introduce environment domains that are included in JEDAI.Ed and conclude with details about the user study and analysis of the results. The large size of the source code prohibits us from including it in the supplementary material, but we plan on releasing JEDAI.Ed as an open source software in case of acceptance (as we mentioned in Sec. 3).

A Desiderata

Table 3 provides a contrast between the features of JEDAI.Ed and JEDAI.

A.1 Intuitive user interface

The JEDAI user interface was a static browser window which would display all the user assistance information on to the screen simultaneously. The result was a text dense webpage. Furthermore, JEDAI required the user to manually click a button in order to check the validity of their plans and receive explanations about their errors. To test even incremental changes in their plans, users would have to move the mouse pointer back and forth between the Blockly workspace and the submit button. Finally, JEDAI presents the robot simulation in a separate browser window, further burdening the user with managing multiple browser tabs or windows while solving tasks.

The unmodified screenshots for JEDAI.Ed and JEDAI are presented in Fig. 11 and Fig. 12 respectively.

A.2 UI Improvements

Extensive work has gone into remodelling the interface of JEDAI to be more user-friendly and intuitive. An important motivation behind this was to not overload the user with too much information on the screen and bring all the components, including the simulation window into a single browser window. Optimizing on the screen space available, JEDAI.Ed presents explanations, hints etc. as collapsible text fields. This allows for a high level summary to be visible immediately on the screen along with the option for the user to expand and consume more detailed information if needed. In order to make the connection between an error explanation and the corresponding block clear to the user, JEDAI.Ed also highlights the action block when its corresponding explanation dropdown is selected. Similarly the robot simulation was integrated within the same webpage containing the Blockly workspace and explanation dropdown lists, thus providing a single, compact view port for the user.

A.3 Blockly

Another accessibility feature introduced in JEDAI.Ed is the ability to minimize and maximize blocks. Tasks that require longer plans, the connected blocks quickly overflow beyond the workspace requiring the user to scroll. Minimizing previously added blocks allows the user to be able to add and edit the arguments of the new block they are adding in the

same workspace without needing to scroll down to find the last block each time.

A.4 State Space Display

To facilitate the users' understanding on the environments that they are solving tasks in, JEDAI.Ed introduces a State Space Display. Starting with the initial state of the environment and listing the change after the application of each action in the users submitted plan, the list of all true predicates is displayed, as seen in Fig. 1*f*. Again, in keeping with the objective of not filling up the screen with text dense messages, the list of predicates is hidden by default and the user can expand and collapse the list as they need. In the event of an invalid submitted plan, the display is truncated at the last valid action and the error explanation dropdown is displayed.

A.5 Curriculum generation

JEDAI and JEDAI.Ed both contain a diverse set of domains and problem tasks for users to solve (App. C), however JEDAI does not offer an in-built method which provides a systematic learning journey to the user. In JEDAI, an educator would have to manually design a lesson plan with students sequentially exposed to problems of increasing difficulty, but that would still be a one-size-fits-all approach, without possibility of the curriculum being tailor-made for individual learners' needs. This problem is alleviated by JEDAI.Ed's curriculum design module which automatically adapts to each individual learner and provides them a customized learning path for learning about the capabilities of a robot in an environment. Our curriculum module progressively generates challenging problems of increasing difficulty, while automatically adapting to users performance.

User created plans are continuously evaluated to assess their level of understanding of the various actions. Each plan is broken down into its constituent actions which are sequentially applied starting from the initial state. An action being applicable in the current state implies the user having a correct understanding of said action. Here, we increment the cost associated with that action and update the current state to be the state resulting in the application of the action on the current state. Conversely, if an action is inapplicable, the entire plan is rendered invalid and updating the current state is impossible. The associated cost of the action is decremented and the action-cost mapping is returned. This above process is illustrated in Alg. 1.

Alg.2 showcases our overall process for adaptive problem generation. To generate a problem task using this action-cost mapping, we use a simple greedy search over the state space, where nodes are states of the environment and weighted edges are actions with the weight of the edge being the cost associated with that action. The fringe is initialized as a minimum priority queue, with the priority of a node set as the cumulative action cost needed to reach that node and the order of addition to the fringe used to break ties. Nodes are then popped from the fringe and for each popped node, the states reachable from that node in one action are added to

the fringe with the incremented cumulative action-cost. This procedure is carried out until an action currently unknown to the user is seen, or if a preset maximum depth is reached. For our experiments, the maximum depth was set to four.

Algorithm 2: Adaptive Curriculum Task Generator

Input: adaptive action-cost mapping A , initial state s_0 , set of grounded actions \mathcal{A} , maximum tree depth d_{max}

Output: Problem Goal State g

```

1  $f \leftarrow \text{Min-Priority-Queue}$ 
2  $v \leftarrow \text{Empty visited set}$ 
3  $\text{add}(f, (s_0, 0, 0))$ 
4  $\text{add}(v, s_0)$ 
5 while True do
6    $s, c, d \leftarrow \text{pop}(f)$ 
7   foreach  $a \in \mathcal{A}$  do
8     if  $\text{isApplicable}(s, a)$  then
9        $s' \leftarrow \text{applyAction}(s, a)$ 
10       $c' \leftarrow c + A[a]$ 
11       $d' \leftarrow d + 1$ 
12      if  $A[a] = 0$  or  $d' \geq d_{max}$  then
13         $g \leftarrow s$ 
14        return  $g$ 
15      if  $s' \notin v$  then
16         $\text{add}(v, s')$ 
17         $\text{add}(f, (s', d', c'))$ 

```

All action costs are initialized uniformly to zero, representing that the user is unfamiliar with all the actions. As the user solves curriculum generated problems correctly, they are presented with problems that each introduce an additional unknown action. This ensures that while solving a problem, the user has only one new action to learn thus easing the cognitive burden placed upon them. The very first curriculum task generated, with all action costs set to zero, will select any action applicable in the initial state and return the state resulting from its application as the goal state. Future work may investigate how this cold start problem impacts user engagement and learning, and how it may be resolved.

A.6 Intuitive Explanations and Hints

In explaining action failures, JEDAI relied upon hand coded text templates, which print out the failing action and the unmet preconditions of said action. While logically sound and providing complete information, these explanations are not very human friendly to read. JEDAI.Ed uses GPT-3.5 to convert these explanations to be more human readable and user friendly. JEDAI is also limited to explaining failure for only one action, which is a handicap JEDAI.Ed does not suffer from.

JEDAI.Ed also allows users to get hints to help them in high-level planning. The hint is displayed in the form of a grounded high-level action, with the action name visible, but

some of the grounded objects that are arguments to the action obscured. This achieves the twin objectives of nudging the user in the right direction while also not spoon feeding them the answer directly.

Hint Generation: JEDAI.Ed comes equipped with FF (Hoffmann 2001), a fast high-level planner. FF can be used to generate the next high-level instruction needed to accomplish the task. This is presented as a user-interpretable hint to the user. Since high-level planning is a hard problem (Bylander 1994), hinting comes preconfigured with a timeout and displays a status message if hints cannot be computed within the time limit.

A.7 LLM Prompts

We use LLMs to translate explanations and hints to be more readable and user friendly. The explanation or hint generated from the user assistance module is augmented with domain and problem pddl files of the problem being solved. This information, along with a simple prompt is fed into the LLM which is tasked with a translating the explanation given the context to a readable natural language description. The LLM itself does not generate any explanation or hint, but merely acts as a translation interface.

GPT-3.5 was prompted with the following text to generate user-friendly explanations and hints, which is domain agnostic and through repeated experimentation, was found to generalize across multiple problems and domains. Text within angular braces acts as a placeholder for the actual text that is input to the prompt, but omitted here for brevity. Note that the LLM is not being tasked with generating any explanation, but only to convert them to natural language:

Explanation Generation Prompt

The following lines describe the $\langle \text{domain} \rangle$ domain file : $\langle \text{domain pddl} \rangle$

The problem to be solved is described in pddl format as: $\langle \text{problem pddl} \rangle$

While running a plan for a problem, an action failed and an explanation generator was used to generate the following explanation:
Explanation: $\langle \text{explanation} \rangle$

The state of the problem - which means the set of predicates that are true in the plan upto the first invalid action are as follows
State: $\langle \text{state} \rangle$

Can you please convert the explanation into a brief, more non-expert friendly message that a novice user can understand? Also, can you suggested briefly what could be done to fix the issue, taking into account the state reached by the plan so far?

The error explanation for a failing action, generated using fixed text templates, as seen in Fig. 1h (marked as the TLDR

version) is presented below:

Text Template Based Explanation

The action at step 3 (Place at location 'counter' object 'can_blue' using gripper 'gripper' this robot 'fetch') could not be performed because 'gripper' is not holding 'can_blue'.

Using the explanation generation prompt we mentioned earlier, and filling in the above text template generated explanation, GPT-3.5 generated the following user friendly explanation, which is also depicted in Fig. 1*h* (marked Explanation) :

LLM Generated Explanation

Explanation: The action of placing the blue can on the counter failed because the gripper is not holding the blue can. To fix this issue, you can have the robot fetch the blue can and place it on the counter using the gripper properly before trying to place it on the table. Make sure the gripper is holding the correct object before attempting to place it at the desired location.

A.8 Prompt Engineering

Educators can customize LLM descriptions for audiences of different expertise by simply designing appropriate prompts and using them as input to JEDAI.Ed's fine-tuned LLM without any additional training. This allows educators to take into account the needs of different users without having to expend resources for training or fine-tuning an LLM. An example prompt that can be used for generating explanations for novice users is stated below.

Hint Generation Prompt

This is the pddl domain file for the <domain> domain :
<domain pddl>

A user has to solve a this problem task described in pddl
<problem pddl>

The plan was run till the problem reached this state - that is the set of predicates that are true :
<state>

And the hint generated, which suggests which next action to take with certain arguments to actions replaced with ? is given below:
<hint>

Can you please convert the explanation into a brief, more non-expert friendly message that a novice user can understand? Also, can you suggested briefly what could be done to fix the issue, taking into account the state reached by the plan so far?

A sample hint generated using hand-coded text templates is illustrated below:

Sample Hint

You might want to try the action: Move To Counter from this location ? the robot ? to this location counter

The LLM generated counterpart is shown below. In a fashion similar to the explanation translation, the LLM here is used to translate and paraphrase the hints and solutions using the hint, domain and problem pddl and the state of the environment passed as context:

LLM Generated Hint

The problem is to solve a task in the Coffee Shop domain. The current state of the problem is that the gripper is empty, there are orders for different cans at the counter location, and the robot is at the starting point. The goal is to place the red can on the red table. To solve the problem, the next action that should be taken is to move the robot to the counter location. This action requires specifying the current location of the robot and the counter location. To fix the issue, you need to execute the move-to-counter action with the appropriate arguments. By doing this, the robot will move to the counter location and be ready to perform further actions to fulfil the goal.

B Modular and Customizable Design

JEDAI.Ed was consciously designed to be modular and customizable, allowing for easy upgrading or replacement of a number of components to suit particular educator needs. We aim to release JEDAI.Ed as an open source software like its predecessor, and hope that the community will build upon and improve our work in order to further the goal of making AI education more accessible to the general public. Despite the many improvements and feature additions in JEDAI.Ed, it is still a lightweight web-based interface which can run on any device with a modern browser. We have also containerized the software to allow the platform to be run on any machine without the user having to worry about installing dependencies. Containerization also assists in deployment over cloud based services allowing for convenient scaling as required, especially suited to schools and other educational settings.

B.1 New Domains and Problems

Adding new domains simply requires the additional environment description dae files, the action configuration specifications of the domain, a semantic mapping of predicates and actions to natural language, and the domain and problem pddl files. JEDAI.Ed given all these inputs handles the creation of the blockly interface, web frontend, setting up the simulator as well as explanation generation on its own. Incorporating new problems within existing domains is even simpler and requires only the addition of new pddl problem files, or can be generated automatically using the curriculum generation module.

B.2 LLMs

There is flexibility in choosing the LLM used in translating explanations and hints to a user-friendly format. JEDAI.Ed can be used with any LLM, since the interface of passing the prompt abstracts away the internals of the explanation generation. The LLM being used may be stored locally in the system or accessed via an API. Educators, therefore, can use LLMs fine-tuned specifically for the task of converting predicates into human readable language, for instance. Further, the prompt being used to generate responses from the LLM can also be modified as required. different prompts can be used to generate responses in a specific format, or to be less or more verbose as needed.

B.3 Hinting

Educators can make hints more, or less transparent to the students as they need. A single tunable real number parameter between 0 and 1 represents the independent probability of each grounded parameter in the action hint being displayed to the user. By increasing this value, hints are more likely to reveal the grounded parameters input to the action in the hint, and vice-versa.

B.4 Simulator

In order to be simulator-agnostic, JEDAI.Ed separates the simulator from the rest of the backend software, and streams the output to the webpage. Our work uses OpenRave streamed using noVNC (noVNC 2024), but any robot simulator package can be used in its place.

C JEDAI.Ed Bundled Environment Details

Even though the educators can add custom environments to JEDAI.Ed, it comes preconfigured with a few environments to help educators. We have seen one such environment in Fig. 1 in the main paper: Fetch robot in the Coffee Shop environment task with delivering orders to various tables. We briefly introduce three more environments here:

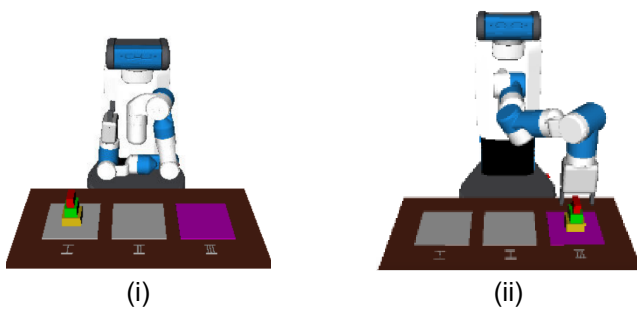


Figure 8: Screenshot of the simulator from the Tower of Hanoi environment. (i) depicts the initial state whereas (ii) a sample goal image for this environment.

Tower of Hanoi In this environment shown in Fig. 8, Fetch robot has to solve the classical tower of Hanoi problem. It consists of three blocks on top of each other kept at a location. There are three such locations. All three blocks

have to be transported to another fixed location, but the robot cannot place a larger block on top of a smaller block. The blocks are uniquely identified using their size.

Keva Planks In this environment shown in Fig. 9, YuMi, a dual-armed robot has to create structures using numbered planks kept on the table. Different structures can be created by placing planks vertically, horizontally or along their edges. Planks can be placed on the table, or on top of other planks.

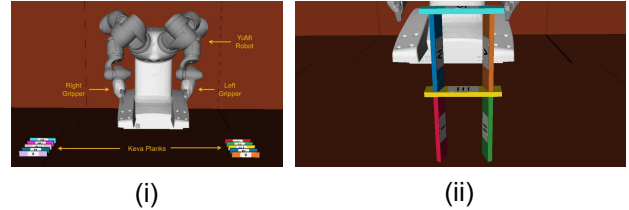


Figure 9: Screenshot of the simulator from the keva environment. (i) depicts the initial state whereas (ii) a sample goal image for this environment.

Dominoes In this environment shown in Fig. 10, similar to the Keva planks setup, YuMi robot has to create structures using dominoes. Each domino can be uniquely identified using an ID that is printed on each domino. The users can move around the camera in the simulator to view these IDs.

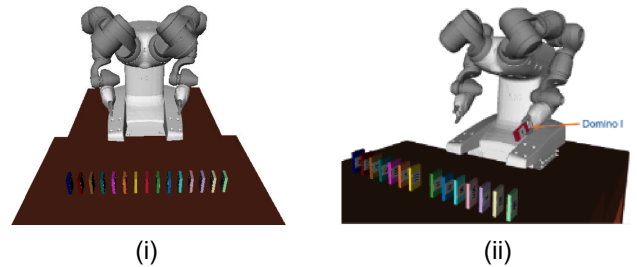


Figure 10: Screenshot of the simulator from the dominoes environment. (i) depicts the initial state whereas (ii) a sample goal image for this environment.

D Study Details

As mentioned in the main paper, the study was divided into four phases. This section provides complete data for the results (Fig. 4, Table 2) presented in the main paper.

D.1 Pre-survey Phase

Table 4 shows the questions and collected responses from the pre-survey questionnaire that was administered as a part of this phase. Note that the participants were yet to interact with JEDAI.Ed or JEDAI in this phase. We provide the statistical significance test results from both the overall participant pool and also segregate the responses of the participants who were assigned JEDAI.Ed or JEDAI in the next

Desiderata	JEDAI.Ed	JEDAI	Reference
Open source	✓	✓	B
Minimal system requirements	✓	✓	B
Highly customizable	✓	✓	B
Integrated simulation	✓	✓	A.2
User-adaptive curriculum problem generation	✓	✗	A.5
LLM translated explanations	✓	✗	A.6, A.7
Hint messages	✓	✗	B.3
LLM translated hint messages	✓	✗	A.6
State space annotation	✓	✗	A.4
Continuous plan checking and explanation generation	✓	✗	A.1, A.2
Explanation of multiple failing actions	✓	✗	A.7
Collapsible information fields	✓	✗	A.1, A.2
Minimize/Maximize blocks	✓	✗	A.3
Clicking on error message highlights corresponding action block	✓	✗	A.2
Robot simulator in the same window	✓	✗	A.1, A.2

Table 3: A comparison of the features of JEDAI.Ed compared to JEDAI.

phase. Our results (one-sample t-test) show that participant responses are statistically significant across these dimensions ($\mu_0^1 = 2$) except for Pre Q5. For Pre Q5, both the combined and per-system statistical results are insignificant. Thus, participants found it neither easy nor difficult to program robots. Similarly, we cannot reject the null hypothesis from the results of the two-sample t-test (that compares responses between JEDAI.Ed and JEDAI for the questions) thus showing that participants assigned to either JEDAI.Ed or JEDAI had similar demographics in our study.

Our pre-study results also show that participants had little to no experience with robotics programming (Pre Q6) and $\geq 50\%$ think about AI (Pre Q2) or use AI-based tools (Pre Q3) daily. This indicates the greater need to impart AI education to people who will be interacting with AI systems frequently.

D.2 Training Phase

There was no questionnaire administered to the participants at the end of this phase.

D.3 Test Phase

Table 5 and Table 6 present the results obtained after administering the post-survey questionnaire at the end of this phase. Depending upon the control group to which they were assigned, participants had interacted with JEDAI.Ed or JEDAI (but not both) at this point. We report these results pictorially in Fig. 4 of the main paper. Post Q1, Q3, Q9, and Q10 are not discussed in the main paper. The results for Post Q1 are not statistically significant. This is not surprising since both JEDAI.Ed and JEDAI manage to increase user understanding of the robot’s limitations and capabilities (Post Q8, H6). We also report results on performing single-tailed two-sample t-tests to analyze responses across the control groups. We use the null hypotheses of no difference ($\mu = 0$) to analyze these results and focus on only a single tail that allows us to analyze whether users prefer JEDAI.Ed over JEDAI. We report the p-values obtained from these

tests. Statistically significant responses ($p < 0.05$) indicate that users preferred JEDAI.Ed over JEDAI in their response.

Table 6 contains JEDAI.Ed specific questions (Q3, Q9, Q10) meant to evaluate the results of the hinting feature of JEDAI.Ed. We did not administer these questions to participants interacting with JEDAI since IRB protocol forbids the exposure of questions that can allow users to infer the presence of a different control group. Our results show that users the test problem moderately challenging and utilized hints only a few times. We attribute this to the training phase where the adaptive task generation algorithm taught the users appropriately such that their need for hints was reduced. Since there was no study administered during the training phase we do not have any data pertaining to the usage of hints during training.

In Table 5, Post Q2 and in Table 6 Post Q10 provided students with a sixth option for the users to choose - "I did not encounter any error" and "I did not receive any "Hints"" respectively. These options were provided to handle the edge cases that the users might experience where for Post Q2, the user do not make a mistake while completing the given problem and for Post Q10, the user did not use the "Get A Hint" button of the interface to get a hint for next step. In Table 5, there is one student who chose the sixth option in Post Q2 after interaction with JEDAI as the first system. In Table 6, seven users opted for sixth option in Post Q10 after interacting with JEDAI.Ed as their first system. These data points for the respective questions were not considered for the statistical analysis of the aforementioned questions because this options do not adhere to the Likert scale that we used for the analysis.

D.4 Sentiment Change Phase

Table 7 and Table 8 presents the overall Sentiment Change that we observed from the data collected from the user study. These results were obtained after the users were asked to interact with second system (S_2). Table 7 contains the overall sentiment change (both positive and negative) for users who

were given JEDAI.Ed as their first system(S_1) and JEDAI as second system(S_2). Table 8 contains the results for sentiment change (both positive and negative) for users who interacted with JEDAI first and then with JEDAI.Ed.

We don't report the paired t-test values for these groups because for most of the questions, the paired distribution does not satisfy the normality assumption of paired t-test. We tested the normality of the data by performing d'Agostino-Pearson test (Pearson, D'Agostino, and Bowman 1977). We were unable to run Wilcoxon Signed-Rank (Wilcoxon 1945) test because the paired data has too many ties and the sample size was insufficient to run the test.

As discussed in section D.3, Post Q2 in Table 7 and Table 8 have sixth option provided to the users that does not adhere to Likert scale. Also, as per IRB protocol, users can choose not to respond to any of the questions in the survey. For both some questions in both the mentioned tables, there are users who have either opted for option sixth in Post Q2 or have chosen not to respond to some questions. In both the cases, we did not consider the data points where either the user has opted for the sixth option or has chosen not to respond as both scenarios differ from Likert scale options that we have chosen for the analysis of the questions. In Table 7, for JEDAI - Post Q1, 1 user has no response and for JEDAI Post Q2, 4 users have opted for sixth option. In Table 8, for JEDAI.Ed Post Q2, 2 users have opted for sixth option.

Question	Question and Responses					t-tests			
						One-sample			Two-sample
						μ_0^1	μ	p	p
Pre Q1	How familiar are you with Computer Science and Artificial Intelligence (A.I.)?								
	Not at all (0)	Slightly well (1)	Moderately well (2)	Very well (3)	Extremely well (4)				
Total	3	14	18	6	1		1.71 ± 0.89	$1.56e-15$	$0.36e-00$
JEDAI.Ed	3	5	8	4	1	0	1.76 ± 1.09	$3.80e-07$	
JEDAI	0	9	10	2	0		1.66 ± 0.65	$2.46e-10$	
Pre Q2	How often do you think about A.I. in your day-to-day life??								
	Never (0)	Once a week (1)	2-3 times a week (2)	4-6 times a week (3)	Daily (4)				
Total	1	3	8	11	19		3.04 ± 1.08	$8.61e-08$	$0.58e-01$
JEDAI.Ed	1	0	3	4	13	2	3.33 ± 1.06	$6.44e-06$	
JEDAI	0	3	5	7	6		2.76 ± 1.044	$0.16e-02$	
Pre Q3	How often do you interact with tools that use A.I.?								
	Never (0)	Once a week (1)	2-3 times a week (2)	4-6 times a week (3)	Daily (4)				
Total	2	4	5	10	21		3.04 ± 1.20	$4.41e-14$	$0.81e-01$
JEDAI.Ed	0	2	2	4	13	1	3.33 ± 1.01	$6.71e-10$	
JEDAI	2	2	3	6	8		2.76 ± 1.33	$3.36e-06$	
Pre Q4	How curious are you to learn about the extent to which A.I. systems and robots can be used today?								
	Not curious (0)	Slightly curious (1)	Moderately curious (2)	Very curious (3)	Extremely curious (4)				
Total	1	4	3	21	13		2.97 ± 0.99	$1.47e-07$	$0.35e-00$
JEDAI.Ed	0	2	1	12	6	2	3.04 ± 0.86	$1.95e-05$	
JEDAI	1	2	2	9	7		2.90 ± 1.13	$0.15e-02$	
Pre Q5	Assume you want a household robot to get you water from the refrigerator. How difficult do you think it is to give it instructions to perform this task?								
	Very difficult (0)	Slightly difficult (1)	Neither difficult nor easy (2)	Slightly easy (3)	Very easy (4)				
Total	3	20	6	10	3		2.23 ± 1.12	$0.17e-00$	$0.93e-01$
JEDAI.Ed	2	7	4	5	3	2	2.00 ± 1.26	$1.00e-00$	
JEDAI	1	13	2	5	0		2.47 ± 0.92	$0.29e-01$	
Pre Q6	How familiar are you with robotics programming?								
	Not at all (0)	Slightly familiar (1)	Moderately familiar (2)	Very familiar (3)	Extremely familiar (4)				
Total	26	13	3	0	0		0.45 ± 0.63	$3.59e-05$	$0.26e-00$
JEDAI.Ed	11	9	1	0	0	0	0.52 ± 0.60	$0.71e-03$	
JEDAI	15	4	2	0	0		0.38 ± 0.66	$0.16e-01$	

Table 4: Pre-survey questionnaire administered during the *pre-survey phase* of our user study (μ_0^1 represents the null hypothesis mean used to conduct the one-sample t-test). We used $\alpha = 0.05$ for determining statistical significance for the t-tests. For each Question ID, the first row is the question as it was presented to the participants. The second row lists the possible answers for the question (and the corresponding Likert scale values in parentheses). The third row presents the responses by all participants. The breakdown of the total responses by control group are presented in the fourth and fifth rows.

Question	Question and Responses					t-tests		
						One-sample		Two-sample
						μ	p	p
Post Q1	After interacting with the JEDAI.Ed system, how inclined are you to learn how daily problems are being solved with A.I.?							
	Not inclined (0)	Slightly inclined (1)	Moderately inclined (2)	Very inclined (3)	Extremely inclined (4)			
JEDAI.Ed	0	2	6	6	7	2.85 ± 1.01	$5.52e-08$	0.14e-00
JEDAI	2	3	4	11	1	2.28 ± 1.10	$3.11e-05$	
Post Q2	How helpful were the explanations that were given for the cause of an error?							
	Not helpful (0)	Slightly helpful (1)	Moderately helpful (2)	Very helpful (3)	Extremely helpful (4)			
JEDAI.Ed	0	1	2	6	12	3.38 ± 0.86	$2.23e-07$	0.67e-02
JEDAI	1	4	5	5	5	2.42 ± 1.20	$0.59e-01$	
Post Q4	How intuitive was the interface?							
	Not intuitive (0)	Slightly intuitive (1)	Moderately intuitive (2)	Very intuitive (3)	Extremely intuitive (4)			
JEDAI.Ed	0	1	6	12	2	2.27 ± 0.71	$9.41e-05$	0.22e-01
JEDAI	0	4	11	4	2	2.19 ± 0.87	$0.16e-00$	
Post Q5	As compared to before participating in this user study, how much has your curiosity increased to learn more about AI systems and robots?							
	Highly decreased (0)	Slightly decreased (1)	Neither increased nor decreased (2)	Slightly increased (3)	Highly increased (4)			
JEDAI.Ed	0	0	1	9	11	3.47 ± 0.60	$4.24e-10$	0.28e-01
JEDAI	0	1	5	8	7	3.00 ± 0.89	$5.17e-05$	
Post Q6	How well do you think you now understand how one can use an AI system to make a plan for a robot to perform a task?							
	Not well (0)	Slightly well (1)	Moderately well (2)	Very well (3)	Extremely well (4)			
JEDAI.Ed	0	1	5	11	4	2.85 ± 0.79	$3.81e-05$	0.46e-01
JEDAI	1	4	5	9	2	2.33 ± 1.06	$0.83e-01$	
Post Q7	Do you agree that the JEDAI.Ed system made it easier for you to provide instructions to a robot for performing tasks?							
	Strongly Disagree (0)	Disagree (1)	Neither Agree nor Disagree (2)	Agree (3)	Strongly Agree (4)			
JEDAI.Ed	0	0	1	9	11	3.47 ± 0.60	$4.24e-10$	0.23e-01
JEDAI	0	1	3	11	6	3.04 ± 0.80	$7.81e-06$	
Post Q8	Do you agree that JEDAI.Ed helps improve the understanding of the robot's limitations and capabilities?							
	Strongly Disagree (0)	Disagree (1)	Neither Agree nor Disagree (2)	Agree (3)	Strongly Agree (4)			
JEDAI.Ed	0	0	2	12	7	3.23 ± 0.62	$1.56e-08$	0.64e-01
JEDAI	0	1	4	12	4	2.90 ± 0.76	$2.78e-05$	

Table 5: Post-survey questionnaire administered during the *test phase* of our user study ($\mu_0^1 = 2$ for the one-sample t-test). We used $\alpha = 0.05$ for determining statistical significance for the t-tests. For each Question ID, the first row is the question as it was presented to the participants. The second row lists the possible answers for the question (and the corresponding Likert scale values in parentheses). Third and fourth row represent the number of participants who chose that answer for the question after JEDAI.Ed and JEDAI respectively. We include here only those questions that were presented to both control groups.

Question	Question and Responses					One-sample t-test	
						μ	p
Post Q3	How challenging were the problems in the JEDAI.Ed session?						
	Not challenging (0)	Slightly challenging (1)	Moderately challenging (2)	Very challenging (3)	Extremely challenging (4)		
JEDAI.Ed	3	10	6	1	1	1.38 ± 0.97	0.88e-01
Post Q9	How often did you use the “Hint” button during the hands-on session?						
	Once per problem (0)	2-4 times per problem (1)	Once during the session (2)	Never (3)	Didn’t notice any hint button (4)		
JEDAI.Ed	4	3	9	5	0	2.23 ± 0.99	0.28e-00
Post Q10	How well do you think “Hint” functionality helped you while solving the problems?						
	Not well (0)	Slightly well (1)	Moderately well (2)	Very well (3)	Extremely well (4)		
JEDAI.Ed	0	4	5	3	2	2.21 ± 1.05	0.45e-00

Table 6: Post-survey questionnaire administered during the *test phase* of our user study ($\mu_0^1 = 2$ for the one-sample t-test). We used $\alpha = 0.05$ for determining statistical significance for the t-tests. For each Question ID, the first row is the question as it was presented to the participants. The second row lists the possible answers for the question (and the corresponding Likert scale values in parentheses). The third row represents the number of participants who chose that answer for the question after interacting with JEDAI.Ed. Since JEDAI does not include hints, per IRB protocol, these questions were excluded from users who interacted with JEDAI immediately before being administered the survey (elaborated in App. D.3).

Question	Question and Responses					Total Sentiment	
						Positive	Negative
Post Q1	After interacting with the JEDAI.Ed system, how inclined are you to learn how daily problems are being solved with A.I.?						
	Not inclined (0)	Slightly inclined (1)	Moderately inclined (2)	Very inclined (3)	Extremely inclined (4)		
S_1 =JEDAI.Ed	0	2	6	6	7	13	2
S_2 =JEDAI	0	5	5	3	7	10	5
Post Q2	How helpful were the explanations that were given for the cause of an error?						
	Not helpful (0)	Slightly helpful (1)	Moderately helpful (2)	Very helpful (3)	Extremely helpful (4)		
S_1 =JEDAI.Ed	0	1	2	6	12	18	1
S_2 =JEDAI	0	3	3	5	6	11	3
Post Q4	How intuitive was the interface?						
	Not intuitive (0)	Slightly intuitive (1)	Moderately intuitive (2)	Very intuitive (3)	Extremely intuitive (4)		
S_1 =JEDAI.Ed	0	1	6	12	2	14	1
S_2 =JEDAI	0	1	11	7	2	9	1
Post Q5	As compared to before participating in this user study, how much has your curiosity increased to learn more about AI systems and robots?						
	Highly decreased (0)	Slightly decreased (1)	Neither increased nor decreased (2)	Slightly increased (3)	Highly increased (4)		
S_1 =JEDAI.Ed	0	0	1	9	11	20	0
S_2 =JEDAI	0	0	1	11	9	20	0
Post Q6	How well do you think you now understand how one can use an AI system to make a plan for a robot to perform a task?						
	Not well (0)	Slightly well (1)	Moderately well (2)	Very well (3)	Extremely well (4)		
S_1 =JEDAI.Ed	0	1	5	11	4	15	1
S_2 =JEDAI	1	4	4	8	4	12	5
Post Q7	Do you agree that the JEDAI.Ed system made it easier for you to provide instructions to a robot for performing tasks?						
	Strongly Disagree (0)	Disagree (1)	Neither Agree nor Disagree (2)	Agree (3)	Strongly Agree (4)		
S_1 =JEDAI.Ed	0	0	1	9	11	20	0
S_2 =JEDAI	1	0	2	11	7	18	1
Post Q8	Do you agree that JEDAI.Ed helps improve the understanding of the robot's limitations and capabilities?						
	Strongly Disagree (0)	Disagree (1)	Neither Agree nor Disagree (2)	Agree (3)	Strongly Agree (4)		
S_1 =JEDAI.Ed	0	0	2	12	7	19	0
S_2 =JEDAI	1	0	0	13	7	20	1

Table 7: Post-survey questionnaire administered during the *sentiment change phase* of our user study. For each Question ID, the first row is the question as it was presented to the participants. The second row lists the possible answers for the question (and the corresponding Likert scale values in parentheses). Third and fourth row represent the number of participants who chose that answer for the question after interacting with JEDAI.Ed and JEDAI respectively.

Question	Question and Responses					Total Sentiment	
						Positive	Negative
Post Q1	After interacting with the JEDAI.Ed system, how inclined are you to learn how daily problems are being solved with A.I.?						
	Not inclined (0)	Slightly inclined (1)	Moderately inclined (2)	Very inclined (3)	Extremely inclined (4)		
S_1 =JEDAI	2	3	4	11	1	12	5
S_2 =JEDAI.Ed	0	3	6	10	2	12	3
Post Q2	How helpful were the explanations that were given for the cause of an error?						
	Not helpful (0)	Slightly helpful (1)	Moderately helpful (2)	Very helpful (3)	Extremely helpful (4)		
S_1 =JEDAI	1	4	5	5	5	10	5
S_2 =JEDAI.Ed	0	1	2	8	8	16	1
Post Q4	How intuitive was the interface?						
	Not intuitive (0)	Slightly intuitive (1)	Moderately intuitive (2)	Very intuitive (3)	Extremely intuitive (4)		
S_1 =JEDAI	0	4	11	4	2	6	4
S_2 =JEDAI.Ed	0	3	6	8	4	12	3
Post Q5	As compared to before participating in this user study, how much has your curiosity increased to learn more about AI systems and robots?						
	Highly decreased (0)	Slightly decreased (1)	Neither increased nor decreased (2)	Slightly increased (3)	Highly increased (4)		
S_1 =JEDAI	0	1	5	8	7	12	1
S_2 =JEDAI.Ed	0	0	4	11	6	17	0
Post Q6	How well do you think you now understand how one can use an AI system to make a plan for a robot to perform a task?						
	Not well (0)	Slightly well (1)	Moderately well (2)	Very well (3)	Extremely well (4)		
S_1 =JEDAI	1	4	5	9	2	11	5
S_2 =JEDAI.Ed	0	4	6	9	2	11	4
Post Q7	Do you agree that the JEDAI.Ed system made it easier for you to provide instructions to a robot for performing tasks?						
	Strongly Disagree (0)	Disagree (1)	Neither Agree nor Disagree (2)	Agree (3)	Strongly Agree (4)		
S_1 =JEDAI	0	1	3	11	6	17	1
S_2 =JEDAI.Ed	0	0	1	14	6	20	0
Post Q8	Do you agree that JEDAI.Ed helps improve the understanding of the robot's limitations and capabilities?						
	Strongly Disagree (0)	Disagree (1)	Neither Agree nor Disagree (2)	Agree (3)	Strongly Agree (4)		
S_1 =JEDAI	0	1	4	12	4	16	1
S_2 =JEDAI.Ed	0	0	5	11	5	16	0

Table 8: Post-survey questionnaire administered during the *sentiment change phase* of our user study. For each Question ID, the first row is the question as it was presented to the participants. The second row lists the possible answers for the question (and the corresponding Likert scale values in parentheses). Third and fourth row represent the number of participants who chose that answer for the question after interacting with JEDAI.Ed and JEDAI respectively.

Execute on Robot
Help
Get A Hint

▼ Actions

- Move robot
- Pick Up
- Place

Start:

Move

the robot: **fetch**

from this location: **starting_point**

to this location: **counter**

Pick up

from this location: **counter**

this object: **can_red**

using this gripper: **gripper**

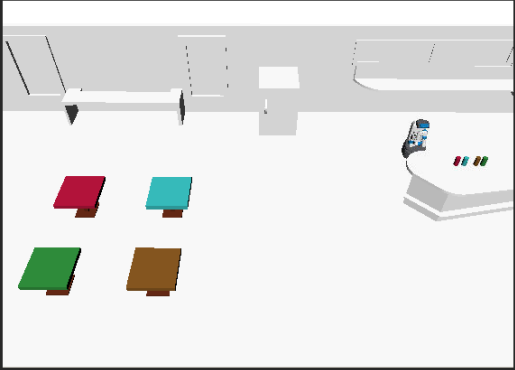
of this robot: **fetch**

Move

the robot: **fetch**

from this location: **counter**

to this location: **table_brown**



Initial State +

After applying Action 1 : Move robot 'fetch' from location 'starting_point' to location 'counter' : +

After applying Action 2 : Pick up from location 'counter' object 'can_red' using gripper 'gripper' this robot 'fetch' : -

can_green is on counter
 can_brown is on counter
 fetch is at location counter
 gripper is holding can_red
 can_blue is on counter

After applying Action 3 : Move robot 'fetch' from location 'counter' to location 'table_brown' : +

Goal not satisfied: Valid plan but does not satisfy goal

'can_green' is not on 'table_green'

'can_red' is not on 'table_red'


Goal State

'can_blue' is on 'table_blue'

'can_brown' is on 'table_brown'

'can_green' is on 'table_green'

'can_red' is on 'table_red'



Expected goal state configuration

Save
Load
Reset
Maximize All Blocks
Minimize All Blocks

Figure 11: Screenshot of the JEDAI.Ed user interface.

JEDAI Explains Decision-Making AI

The screenshot displays the JEDAI user interface, which includes a top navigation bar with buttons for "Submit Plan", "Help", "Open Simulator", and "Stop Execution". On the left, an "Actions" menu lists "Move robot", "Pick Up", and "Place". A "Start:" button is positioned above a red error box containing the text: "Move... the robot: fetch from this location: table_brown".

The central part of the interface features a 3D simulation window titled "OpenRAVE 0.9.0 (Development Version)" in an Incognito browser. The simulation shows a blue and white robot in a room with several tables and chairs. A "Clear Plan" button is located at the bottom right of the simulation window.

Below the simulation window, a dark grey box displays the "Goal State": "can_blue' is on 'table_blue', 'can_brown' is on 'table_brown', 'can_green' is on 'table_green', 'can_red' is on 'table_red'".

On the right side, a red error box states: "BAD ACTION: The action at step 1 (Move robot 'fetch' from location 'table_brown' to location 'counter') could not be performed because 'fetch' is not at location 'table_brown'". Below this error message are two 3D simulation images: "Expected goal state configuration" and "Initial configuration".

Figure 12: Screenshot of the JEDAI user interface.