Using Explainable AI and Hierarchical Planning for Outreach with Robots

Rushang Karia*, Jayesh Nagpal*, Daksh Dobhal*,
Pulkit Verma, Rashmeet Kaur 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 Computer Science/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 curriculum 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 (see Table 1 for more differences). 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*,

*These authors contributed equally.

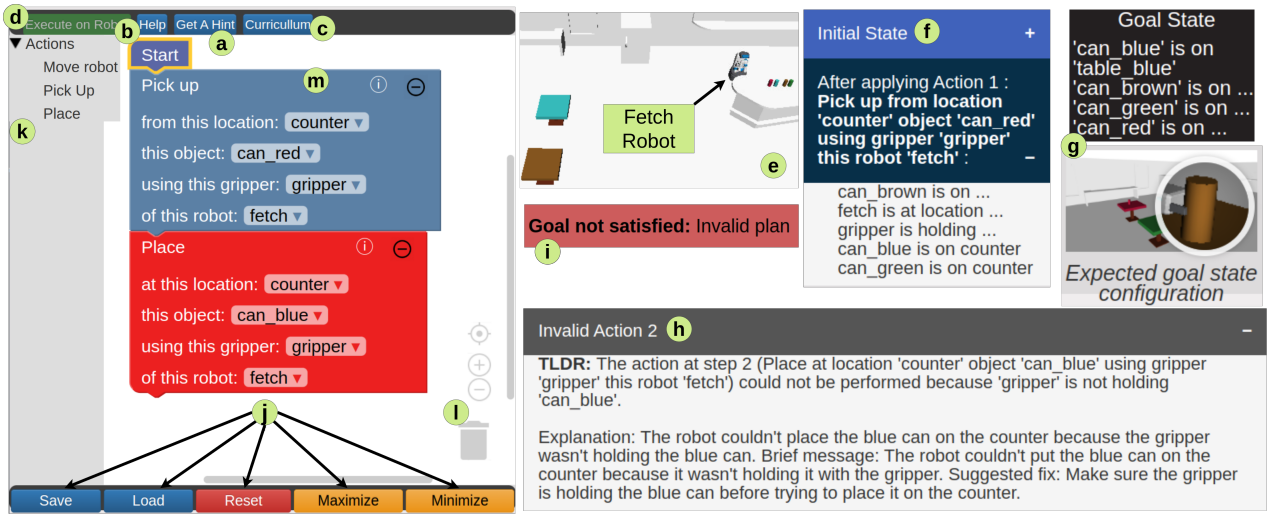

Figure 1: A screenshot (best viewed in color) of the JEDAI.Ed user interface (UI) (zoomed in and enhanced for clarity). The annotated circles describe the different sections of the UI (described in Sec. 3.2). The extended version (Karia et al. 2024) includes unmodified screenshots and the code repository (Fn. 1) contains a video walkthrough of the JEDAI.Ed interface.

planning is the process of computing such plans, and algorithms that do planning are called *planners*. Planners take an 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: $\theta_1, \ldots, \theta_8$. A motion plan, $\langle [\theta_1^1, \ldots, \theta_8^1], \ldots, [\theta_1^n, \ldots, \theta_8^n] \rangle$, 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 $[\theta_1^i, \ldots, \theta_k^i]$ 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, $\langle$`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 $\langle$`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 – described in the extended version (Karia et al. 2024) – to develop JEDAI.Ed, an open source[1] 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, etc.

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

[1]Source code is available at: https://github.com/AAIR-lab/jedai

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 feature comparison between JEDAI.Ed and JEDAI. See the extended version (Karia et al. 2024) for a detailed description.

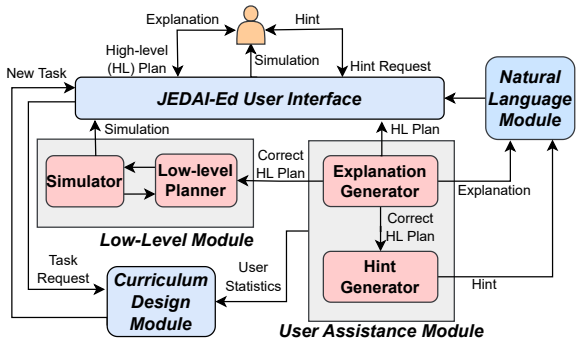

Figure 2: The JEDAI.Ed architecture (described in Sec. 3.2).

robots to perform tasks. Furthermore, our platform explains 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 & Reasoning[2] 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 the extended version). 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. 1k) to create plans via Blockly (Google 2017) – a block-based programming language. For example, the `Pick Up` action in Fig. 1m represents the high-level instruction '*Pick Up the red can from counter using gripper of robot Fetch*'. 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/Motion Planning Module (MPM) JEDAI.Ed provides an integrated low-level planner, ATAM (Shah et al. 2020) and simulator, OpenRAVE (Diankov 2010). ATAM converts high-level plans to low-level plans (Fig. 1d) that can be executed and visualized on the UI via the simulator (Fig. 1e). This execution is a close approximation of the real-world. The planning process is streamed in real-time providing informative insights about it. The video walkthrough in the code repository contains one such execution.

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 explainable 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 extended version.

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 di-

[2]https://ai4k12.org/big-idea-2-overview/

Algorithm 1: Adaptive User-Performance Tracking

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

$s \leftarrow$ getCurrentState()
if *canExecuteAction*(s, a) ***and not*** h **then**
    // User knows action: Cost ↑
    $C_u[a] \leftarrow C_u[a] + 1$
else
    // User does not know action: Cost ↓
    $C_u[a] \leftarrow C_u[a] - 1$

rected fashion. We do so by keeping track of the user's performance 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 extended version.

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 below describes the typical processes involved in a session.

First, the educator installs JEDAI.Ed on a machine. Next, the educator uses the CDM to select an appropriate environment for the students. The educator then generates (or selects preset) tasks for the student to accomplish. Alternatively, the educator could instruct the students to use the adaptive problem generator to generate tasks. 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 using textual and visual descriptions. 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 and connecting them to the *Start* block (Fig. 1m). An audible click lets the user know that the block snapped to another block.

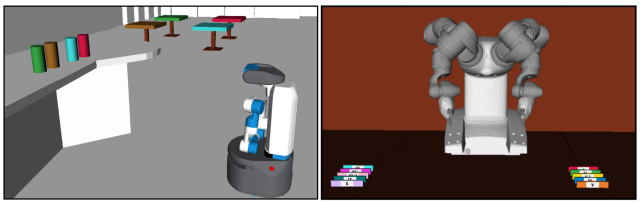

Figure 3: Example environments, Coffee Shop (left) and Keva (right), included with JEDAI.Ed. These environments feature sample tasks and problem generators. Details of other environments are included in the extended version.

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 states resulting from the execution of their current plan in the state display area (Fig. 1f). The user may also request a hint (UAM, Fig. 1a, elaborated in the extended version) 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 "Execute 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. 1e).

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 sample size of 42. There were 23 non-STEM participants.

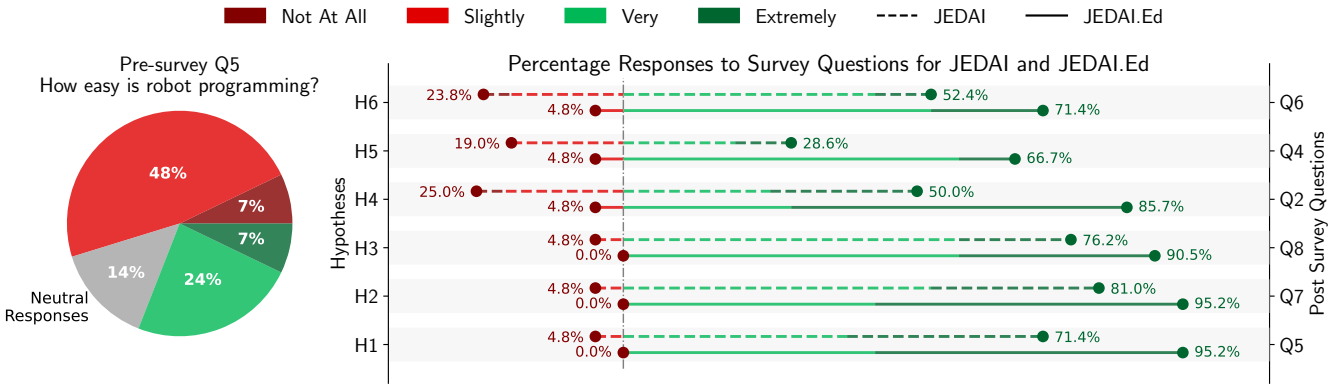

Figure 4: Results from our user study, with $n = 42$, split evenly between JEDAI.Ed and JEDAI. The x-axis plots the responses as a percentage. Green (red) bars to the right (left) indicate positive (negative) sentiment. Stem annotations indicate the total positive (negative) sentiment. The left (right)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

		$\mu \pm \sigma$ **w.r.t. 1-sample t-test** ($\mu_0^1 = 2$)	
Hypothesis	**Post Survey Question used for Testing Hypothesis**	**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 briefly describes 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 extended version (Karia et al. 2024)

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

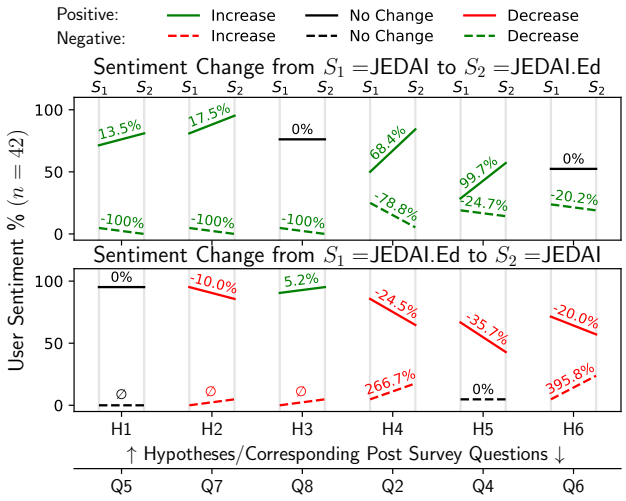

Figure 5: Slope charts for our sentiment change experiments. Users interacted with S_1 first and S_2 next. The y-axis shows the absolute user sentiment while line annotations show the % improvement ($\frac{S_2-S_1}{S_1} \times 100$). We use $\varnothing$ when $S_1 = 0$.

once again asked to answer a post-survey questionnaire. 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

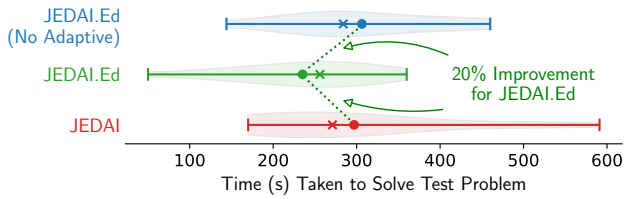

Figure 6: Violin plots that indicate the time needed to solve the test task. • (×) represents the mean (median).

that they better understand the robot's capabilities, and 71% 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 (≈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 occured. 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 tesk 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 key advantage of JEDAI.Ed is the adaptive problem generation that appropriately adjusts the task difficulty to facilitate faster learning. JEDAI.Ed also informs users of invalid actions and explains them in real-time as compared to JEDAI where users need to submit plans to get any feedback.

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 6 sessions at three different high schools to a total of ≈240 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 some of our visits are included in Fig. 7. We also solicited feedback from the program coordinator(s) 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.

We hosted JEDAI.Ed on AWS (Amazon 2023) for our pilot program. The network latency between the school and the server caused issues wherein the video stream of the robot executing the plan was not rendering correctly. Optimizing the motion planner to break down the trajectory and send it piece-by-piece would provide a smoother experience.

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 like tablets etc., 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 multi-modal models.

An additional feature that we are working on allows users to form their plans using programming constructs like loops and conditionals. Explaining failures in such programs is a challenging and exciting question for future research.

Finally, improving the adaptive problem generator to dynamically track progress and perform non-uniform cost increases is an interesting avenue for future work.

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.

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.

Figure 7: Student engagement in JEDAI.Ed's pilot program.

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.

Ethical Statement

This work involved recruiting humans for our study. Both our pre and post survey questionnaires went through an Institutional Review Board (IRB) review process and were approved before starting the study. We opted to recruit university students instead of using our high school pilot program for the study due to the difficulties associated with conducting user studies involving minors. To maintain a similar level of expertise, 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. This resonates well with the demographics of our high school pilot program which was mainly composed of students from grades 9-12.

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.

Acknowledgements

This work was supported in part by the ONR under grant N000142312416 and by the NSF under grant IIS 1942856.

References

Amazon. 2023. Amazon S3. https://aws.amazon.com/. [Online; accessed 15-Aug-2023].

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*.

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. 2017. Blockly. https://github.com/google/blockly. [Online; accessed 5-Sept-2022].

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.

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.

Karia, R.; Nagpal, J.; Dobhal, D.; Verma, P.; Nayyar, R.; Shah, N.; and Srivastava, S. 2024. Using Explainable AI and Hierarchical Planning for Outreach with Robots (Extended Version). arXiv:2404.00808.

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 *Proc. Creativity and Fabrication in Education*.

Kumar, A.; Vasileiou, S. L.; Bancilhon, 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.

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].

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*.

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*.