Proceedings of the Thirty-Second International Conference on Automated Planning and Scheduling (ICAPS 2022)

Actor-Focused Interactive Visualization for AI Planning

Gabriel Dias Cantareira, Gerard Canal, Rita Borgo

Department of Informatics, King's College London
gabriel.dias_cantareira@kcl.ac.uk, gerard.canal@kcl.ac.uk, rita.borgo@kcl.ac.uk

Abstract

As we grow more reliant on AI systems for an increasing variety of applications in our lives, the need to understand and interpret such systems also becomes more pronounced, be it for improvement, trust, or legal liability. AI Planning is one type of task that provides explanation challenges, particularly due to the increasing complexity in generated plans and convoluted causal chains that connect actions and determine the overall plan structure. While there are many recent techniques to support plan explanation, visual aids for navigating this data are quite limited. Furthermore, there is often a barrier between techniques focused on abstract planning concepts and domain-related explanations. In this paper, we present a visual analytics tool to support plan summarization and interaction, focusing in robotics domains using an actor-based structure. We show how users can quickly grasp vital information about actions involved in a plan and how they relate to each other. Finally, we present a framework used to design our tool, highlighting how general PDDL elements can be converted into visual representations and further connecting concept to domain.

Introduction

As with all fields of AI research, the past years saw an increase in adoption of AI Planning technology, leading to an increasing demand in methods to interpret and explain such models. Reasons for this increase include the drive to improve such systems, to increase user trust, and to provide transparency for legal action, especially considering many of those systems will be employed in sensitive applications. AI planning applications are commonly transparent and able to provide direct information regarding how a plan came to be or what are the relationships between individual actions, but directly interacting with this information is often not enough to convey to users the knowledge they need, especially as plans grow larger and more complex and user needs become more specific (Fox, Long, and Magazzeni 2017).

While there has been recent development in the field of Explainable AI Planning (XAIP), approaches are often textual in nature. Although an often seen feedback in user studies is the request for visual aids, very few visual analytics approaches to plan explanation are present in literature. While simple data visualization tools can provide helpful insight, methods to explore and interact with the specific structure of planning problems are currently limited. Additionally, a gap has been reported in XAIP applications between domain-oriented design and approaches based on abstract planning concepts (Lindsay 2020), which can hinder the broader impact of newly-developed explanation technologies.

In this paper, we present a novel visualization tool for AI planning, focused on the domain of robotics. We aim to start building a bridge between domain-oriented and concept-oriented planning explanations, by presenting a model that is still based on certain domain characteristics (highlighting actors and movement) while building a framework that is extensible to general PDDL problems.

In short, the main contributions of this paper are:

- A Visual Analytics tool to explore and interact with PDDL plans;
- A glyph system that enables explanation and summarization of actions.

Related Work

AI Planning Explanations

A concept discussed and employed in several approaches in literature is that an explanation needs to address multiple models with conflicting properties in the same scenario, mostly pertaining to the actual system versus its image in the user's mind. *Model reconciliation* (Chakraborti et al. 2019; Sreedharan et al. 2019) states that the role of an explanation technique is to bridge the gap between model and user's perception of such model by providing the user with knowledge that might be missing. *Model Acquisition* (Sreedharan et al. 2020) aims to move the actual model (*revealed* model) towards an ideal model intended by the designer (*mental* model).

Another common approach in XAIP are *contrastive explanations* (Miller 2018; Hoffmann and Magazzeni 2019; Dhurandhar et al. 2018), which aim to provide knowledge by comparing different possibilities or courses of action. Fox, Long, and Magazzeni (2017) state that, in the context of AI planning, the question "why did you do that" is often loaded with the context "why did you do that, *as opposed to something else*", and an appropriate response should take alternatives into account. By providing information regarding

Copyright © 2022, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

other possibilities, contrastive explanations inform the user of the reasoning behind the choice of a given course of action (Borgo, Cashmore, and Magazzeni 2018; Sreedharan, Srivastava, and Kambhampati 2018).

As methods for generating and storing information intended for explaining planner decisions improve, how to properly communicate this information to users has also become a research focus. *Argumentation frameworks* (Collins, Magazzeni, and Parsons 2019) can be used to build explanations from low-level logic conditions and effects. For robotics in particular, *verbalization* (Rosenthal, Selvaraj, and Veloso 2016; Hayes and Shah 2017; Canal et al. 2022) is a powerful tool that allows a robot to convey argumentation into natural language. On the general field of XAI, there are discussions on how explainability itself contains a social component between explainer and explainee, and how to properly handle it (Madumal et al. 2018; Miller 2019).

Lindsay (2020) argues that there is a gap in the field of XAIP between domain-tailored explanations, which focus on specific problems and applications, and explanations based on general planning concepts, which are more abstract in nature. The author then proposes an explanation method that builds an abstraction for a given problem based on generic problem structures that can be shared between domains. The aforementioned gap is relevant to visualization as well, as the effort involved in developing unique visual representations for each domain can make it impractical. In this paper, we work towards developing a visualization system that can be decomposed into general components, which could then be used for other applications.

Visual Analytics and XAIP

Visual Analytics (VA) is the practice of using visual representations and interfaces to promote analytical reasoning with respect to complex computational systems or data. VA is commonly described as a cycle in which users obtain knowledge about a given model, act on that knowledge to enact change, and then obtain new knowledge from the changed model that can be used for further improvement. While VA approaches are quite common in XAI (Liu et al. 2017; Choo and Liu 2018), the literature on visualization techniques for plan explanation and interaction is currently quite limited.

Visualization approaches to XAIP proposed in literature include Tintarev et al. (2014), who proposed a method to present plans as either text or graph form, allowing alternating between the two representations and using basic exploratory functions such as filtering and highlighting. Magnaguagno, Pereira, and Meneguzzi (2016) proposed a visual representation for planning that depicts actions as jigsaw puzzle pieces, and different PDDL predicates as connector shapes that must be matched for an action's preconditions to be satisfied. While both approaches provide insights into how plan actions relate to one another and how they are organized regarding their preconditions and causal links, they lack context, not showing the role these actions play in a general view of the plan. Additionally, as plans grow larger in size and complexity, navigating through each action and effect individually may quickly become impractical. Kumar et al. (2021) proposed a visual system for connecting actions and representing system states applied to model reconciliation, but it also lacks scalability as plan size and complexity grows larger. Chakraborti et al. (2017) presented a visualization-enabled system for XAIP that offered summarization and reasoning, but mainly for understanding relationships between sensorial data gathered by the system and planner outputs. Plans themselves are shown as a sequence of action nodes, with preconditions and effects shown as arrows coming in and out of action nodes, respectively. Le Bras et al. (2020) presented a timeline-based approach that condenses actor state into the Y axis, aiming to provide insights on how actors may block or interact with each other. Benton et al. (2018) also utilize vertical information in an innovative way, presenting an aircraft landing plan as a Gantt chart in which actions are stacked according to the altitude in which they take place. While these approaches are very effective, vertical positioning alone may not be enough to convey all necessary information for many domains.

The visual representations we present in this paper build upon information visualization concepts in literature. Among them, we can cite glyph design (Borgo et al. 2013), particularly hierarchical glyph grouping (Rees et al. 2020) and taxonomy-based design (Maguire et al. 2012).

Design

This paper presents a novel visualization method for AI plan explanation and summarization in domains related to autonomous robots. Our model is designed to structure plan information in a timeline around different *actors*, individual real-world entities performing actions and controlled by the planner (i.e., robots). The design is based on observation of other plan explanation approaches, such as verbalization: natural languages often form sentences using a subject-verb structure, denoting an actor performing an action.

We researched literature on planning for robotics and developed our model around the most commonly found domain features and problems, as well as user studies. One example of user feedback can be found in (Canal et al. 2022), which showed users two different verbalization approaches and asked for feedback in how effective they were at presenting information. Users had varying levels of knowledge in robotics and planning, and at the end were asked for general feedback regarding suggestions for improvement or what information they thought was missing from the verbalized explanations. The most common requests related to summarization and contextualizing actions, in particular how each action related to goals. More than a third of the participants explicitly asked for visual aids when asked about suggestions to an explanation tool. We consulted robotics experts for questions that a visualization tool would need to answer as well, and refined our results to a list of four requirements and four design tasks that would attend them.

Requirements

R1 **Summarization.** Interacting with longer and more complex plans by reading a list of actions can quickly become unfeasible or excessively time-consuming. Users need a general plan representation able to quickly provide general information on the actions contained within.

R2 **Context.** Besides showing *what* actions are contained within a plan, a XAIP tool must also explain *why* they are there. The role of a given action within a plan can be contextualized by examining which other actions are impacted by it or how it relates to the plan's goals.

R3 **Task-oriented.** A large problem in XAIP is that the questions to be answered often depend on the task at hand. The visualization tool must be able to accommodate to different users, domains and needs.

R4 **Details on demand.** A visual summary can be helpful, but access to detailed, accurate and unambiguous information from data must be provided when requested.

An important note is that such requirements interact with one another; **R1** refers to building representations that can summarize the data contained within, but **R3** states that information shown depends on task. Therefore, the contents of a summary will be different for different tasks. From these requirements, we devised four design tasks:

Design Tasks

T1 **Action Grouping.** Essential for **R1**. The visualization model needs to be able to bundle groups of actions into single entities that represent them. An efficient representation for a group should contain all relevant information (**R2, R3**) from the actions contained within, or provide quick ways to access it (**R4**).

T2 **Action Details.** All information related to a given action, either properties (i.e, parameters, start time or associated PDDL functions) or context (i.e, how they relate to goals, objects and other actions in the plan), must be visually represented (**R2, R3**) or available upon demand (**R4**).

T3 **Goal Tracking.** The visualization should provide an user with understanding of how close goals are to completion along the execution of a plan, when they are completed and the roles of actions with respect to them (**R1, R2**).

T4 **Information Filtering.** Depending on the user and domain, certain types of information need to be prioritized, while others can be ignored. Users need easy access to details that are relevant to their tasks while not having to navigate through all possible detail options (**R3**).

Visualization Framework

The proposed visualization consists of multiple glyphs representing actions arranged horizontally according to a timeline and vertically according to the actor performing them and the plan they appear in. The contents and presentation of each glyph are customizable, allowing for exploratory analysis or for expert users to build a profile for a given domain with a set of behaviors assigned to different types of action.

Tool Overview

The visualization tool was implemented in d3.js[1], and runs on most web browsers. Plan data extraction was done through ROSPlan (Cashmore et al. 2015), and is compatible with PDDL 3.1. A screenshot of the visualization tool is shown in Figure 1. (A) shows file selector and customization options, allowing users to change visualization features for certain operators or objects, as well as choosing an action grouping method. (B) shows different plans stacked vertically for comparison, with the top row being a timeline for external events, such as Timed Initial Literals. (C) shows the message box that appears when the cursor hovers over a glyph, containing detailed information on the action (or group of actions, in the case of a group glyph), and highlights causal links by giving causally related glyphs colored auras. An expandable sidebar on the right (D) contains information on the plans themselves, such as their full description in PDDL form, as well as all goals and their color legend.

Glyph Design

Glyphs can be defined as visual objects that depict attributes of data records using different visual channels and are placed discreetly in display space. They can be used independently and constructively to depict attributes of a data record or a set of data records, and can make use of visual features from other signs such as icons and symbols. (Borgo et al. 2013).

Plan actions can contain several kinds of information, from their operational details (type of operator, parameters) to their relationships to other actions (causal links) and goals (causal chain, goal completion). We define action glyphs as a circle, with operator type shown as a symbol in the middle. Goal information is shown as colored arcs on the outer edge of the circle and dots surrounding it. Temporal information is shown as a bar spanning the action's duration, and an additional sectioned arc surrounding the glyph can be customized to display other variables. Figure 2 shows the proposed glyph and the information it contains.

Icons and hierarchical taxonomy An icon is displayed in each glyph's middle area, representing the type of action (PDDL operator). These icons are selected by users, and the set of icons is used to create a hierarchical taxonomy for actions in that domain (**T1, T2**). This hierarchy is used for grouping actions together and summarizing groups of similar actions. Figure 3 shows an example of taxonomy for the IPC2002 *Rovers* domain (Long and Fox 2003). Operators can belong to four different categories, which are used to group and identify icons in the visualization. This taxonomy can also be used to enable different levels of abstraction for different action types in the same visualization (**T4**): in this example, each action of the 'interaction' type uses a specific icon, but all 'communication' actions share the same icon.

While we originally intended to propose building operator representations using a more abstract taxonomy-based hierarchical system (Maguire et al. 2012), empirical observation showed that a semantically appropriate icon (Mcdougall, Curry, and De Bruijn 1999) can be instantly recognizable, reducing the learning curve that an abstract hierarchical representation would introduce. Although we created the icons shown in this paper using basic shapes, an icon library can be plugged into the tool and made available to users.

[1] https://d3js.org/

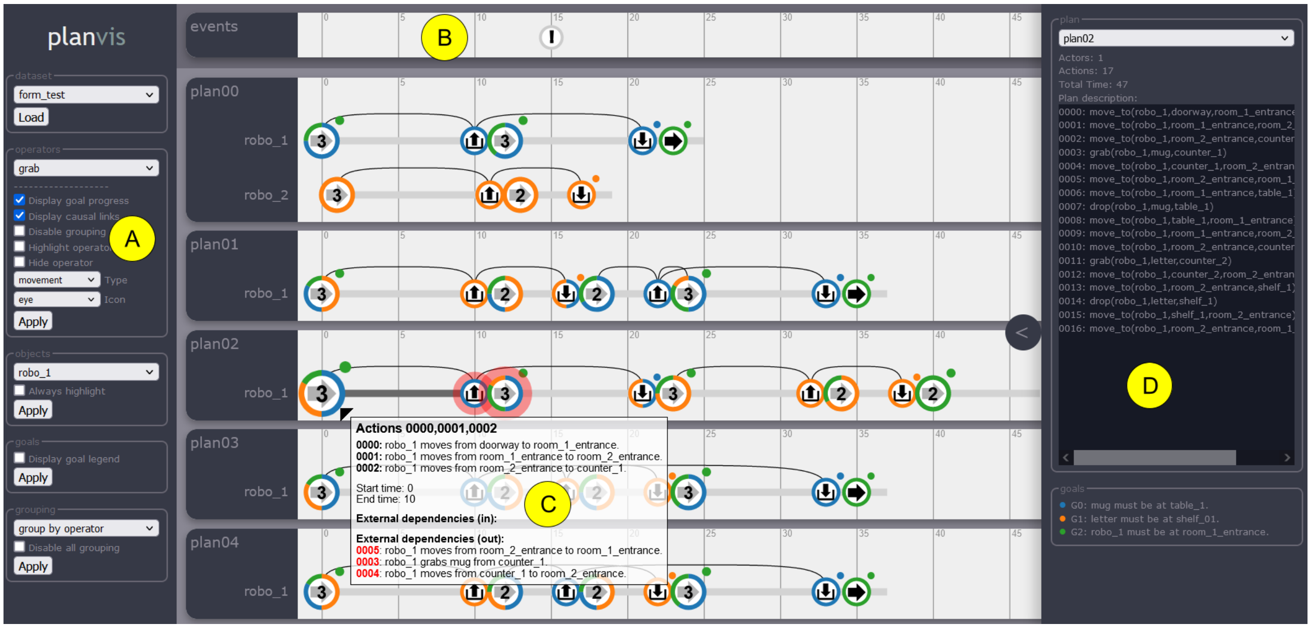

Figure 1: Screenshot of the plan visualization tool. Settings and customization menu is shown on the left (A). The center area shows a timeline shared by a list of plans, distributed vertically (B). When an action glyph is selected, a message box is displayed, containing its details (C). An expandable tab (D) on the right contains detailed information for all plans and goals.

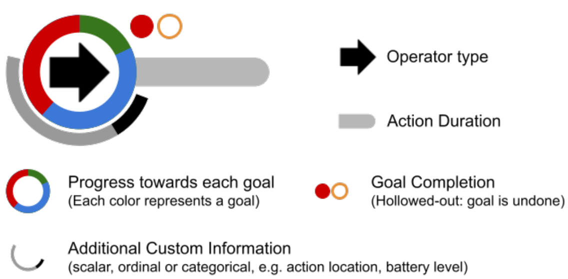

Figure 2: Action Glyph description

Actions	Movement	navigate
	Interaction	sample_soil
		sample_rock
		drop
	Perception	calibrate
		take_image
	Communication	communicate_soil_data
		communicate_rock_data
		communicate_image_data

Figure 3: Example Hierarchical Action Taxonomy for the IPC2002 *Rovers* domain (Long and Fox 2003). Icons and hierarchy are user-defined and are used for grouping actions and assigning behaviors to similar groups.

Goal Relevance Estimation To provide users with insights on the role of a given action in a plan, we produce an estimation of its impact in each of the plan's goals (conditions that need to be true or false by the end of the plan) (**T3**). This is accomplished by estimating how close each goal is to completion before and after the action takes place and comparing them. Given a plan $\pi = \{(t_0, a_0)...(t_n, a_n)\}$ where (t_i, a_i) is a pair of time instant t_i and action a_i, a set of goals $G = \{g_0, ..., g_m\}$, an initial state s_0, a domain D containing the types of actions that can be undertaken, and an external planner algorithm *planner* that returns a generated plan (π_g) and its total duration (c). Using Algorithm 1, we can obtain a $n \times m$ matrix Δ containing the estimated progress for all goals in G and actions in π.

Δ essentially contains, for each action a_i, the predicted difference in time between completing each goal with and without a_i's effects. It allows the observation of the impact of different actions in a plan with respect to each goal, and shows that most actions can be connected to a subset of goals that they bring the system state closer to achieving. In some cases, actions can negatively impact goals (e.g., moving away from where an action needs to take place to fulfill a goal). An example Δ is displayed in Figure 4.

This information is added to the visualization as a multi-colored segmented circle around the action icon in each glyph. The circle is divided into segments, each representing a goal and assigned a different color using a categorical color scale. The fraction of a_i's circle assigned to each color is linearly distributed among the non-negative values of Δ_i. Actions that for whatever reason do not have any positive values in Δ will have no colored outline at all.

Additionally, we mark actions that turn a goal estimate to 0 (action 'completes' goal) and actions that raise goal estimates from 0 (action 'undoes' goal), and add dots to the upper-right corner of glyphs: for each goal completed, a dot of the corresponding goal's color is added, and for each goal undone, a hollowed-out dot is added.

It is worth noting that all information obtained in this step

Algorithm 1: Goal progress estimation

Input: π, G, s_0, D
Output: Matrix of goal advancement per action Δ

Load initial state s_0
let M, M' be $n \times m$ matrices
for each action a_i in π **do**
    S_i = system state immediately after a_i ends
    S'_i = S_i without a_i's effects
    for each goal g_j in G **do**
        $(\pi_{ij}, c_{ij}) = planner(D, S_i, \{g_j\})$
        $(\pi'_{ij}, c'_{ij}) = planner(D, S'_i, \{g_j\})$
        $M(i,j) = c_{ij}$; $M'(i,j) = c'_{ij}$
    end for
end for
$\Delta = M' - M$
return Δ

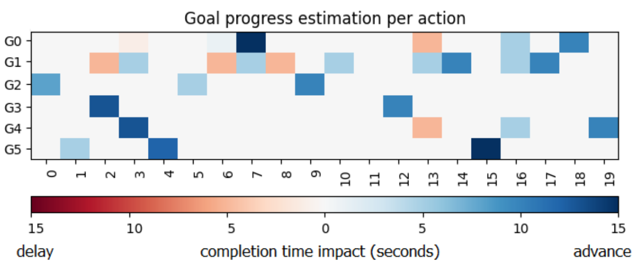

Figure 4: Goal estimation matrix Δ for a plan using the IPC2002 *Rovers* domain. The plan has six goals and 19 actions. Blue cells appear when an action reduced a goal's estimated completion time, and red cells represent a goal completion time increasing, i.e., the effects of the action made the goal more difficult to achieve individually.

are estimates meant to provide insight into the roles of each action in a plan, and are not guaranteed to be an exact representation of its time constraints and dynamics between plans and system states. Isolating and quantifying the effects of individual actions may come with a degree of error, especially as the number of simultaneous actions that can be performed increases. Therefore, we also give the option to turn off goal tracking for chosen operators.

To perform goal progress estimation, the planner can be run using a relatively short timeout limit, as results are designed to be only interpreted as an approximation, and planning for a single goal is less time-consuming. With a 0.25 second timeout, a full analysis for a 100 action plan and 10 goals will take under 5 minutes. A planner able to keep track of states through multiple runs could drastically improve performance as it would obtain many similar plans.

Spatial Layout

The 2D visual space in the visualization is organized as follows: the horizontal (x) axis is reserved for a timeline, that is scaled to the screen by the tool. Glyphs are placed according to start time of their respective actions or action group. The vertical (y) axis is divided among plans, which are displayed in separate areas for quick comparison. Each plan is then further divided between actors, and each action is placed in the

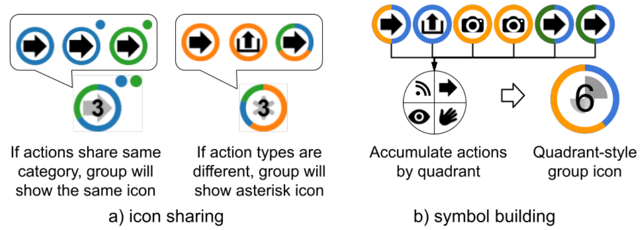

Figure 5: Two approaches to building group glyphs. a) applying the same icon if multiple actions share an operator (or hierarchy branch) or asterisk if there are multiple actions of unrelated types. b) a quadrant symbol based on the number of actions from each hierarchy branch. Other elements, such as goal tracking and duration, remain the same.

row of its main actor. We define the main actor for an action as the first parameter in each action command.

Glyph Grouping

If visual density becomes high enough that neighboring glyphs cannot be displayed side-by-side without overlapping, they are grouped into a larger glyph that represents a group of actions (**T1**). Goal estimation is displayed as if the group was a single action; duration is converted to the first action start until the latest action end. Additionally, to highlight the presence of multiple actions, the group glyph is drawn 25% larger than single-action glyphs.

We set two ways to generate an icon for an action group. The first is to use the same icon if all actions share the same operator, or category icon on the hierarchical taxonomy if they belong to the same category. Otherwise, an asterisk icon is used. The second approach is to build a symbol based on assigning quadrants for higher-level taxonomy categories and filling them (both in size and opacity), either according to the number of actions of the given category or according to the sum of their durations. For both methods, the number of actions in the group is added to the middle of the glyph. Figure 5 illustrates the two approaches to glyph grouping.

Grouping is done by agglomerative hierarchical clustering in actions filtered by actor. By default, distances are calculated between actions' start times on the timeline, and the threshold is automatically defined according to screen resolution, zoom level and glyph size. It can also be triggered manually and configured for other metrics: using the Cosine distance between goal estimates allows combining actions that contribute towards similar goals, and comparing operator types can group actions that share the same type to improve readability. However, to preserve timeline structure, comparisons can only be made between adjacent glyphs.

Interactivity

The visualization is designed to support zooming and panning actions, automatically rebuilding groups if the zoom level causes them to overlap or stop overlapping. When the cursor hovers over a glyph, it is selected, increasing its size and displaying a message box with the full content of the action (**T2, T4**): id, description, start time and duration, causal links. Other glyphs relating to its causal links, i.e., actions that enable the selected action to happen and actions that the selected action enables (Seegebarth et al. 2012), are highlighted in green and red auras, respectively. The same effect applies to group glyphs, with the exception that only causal links external to the group will be listed and highlighted.

Customizable Behavior

It is important that different behaviors can be assigned to different types of glyphs (**T4**), as different actions may have different meanings and thought processes associated with them depending on the task. Therefore, the visualization tool has the option of filtering action glyphs by operator types and parameters, assigning different visual properties to them. For instance, causal links may be of interest only to a certain type of action, and would generate unnecessary visual clutter if shown in others. The visual options available for subsets of actions include highlighting and hiding glyphs, displaying causal links as arcs, disabling goal progress display, and disabling grouping. The visualization tool uses a domain metadata file to save customization options and can include verbalization for describing actions and goals (otherwise described in standard PDDL format). This set of options can either be built as part of exploratory analysis, as an user learns about the domain and problem, or be given as a standard, previously set up by a domain expert and then given to end users to aid in comprehension.

Applications

In this section, we present example applications of the visualization tool, discussing how it can support analysis for problem exploration and planner debugging. In the interest of space, we focus on information given by a few of the visual elements discussed in previous sections.

Problem Exploration In this example, shown in Figure 6, we investigate how objects in the problem interact with a plan, and how they relate to different actions. The plan was generated from a problem using the IPC2002 *Rovers* domain (Long and Fox 2003). There are 4 actors and 11 goals to achieve, all related to the rovers collecting samples and transmitting data to a lander in a different location and optimized for time. A few elements are noticeable at first glance: *rover1* tackles all camera actions (eye icons) and image-related goals. *rover0* has actions that work towards multiple goals, while *rover2* and *rover3* work mainly towards one goal each and have large amounts of downtime. To investigate possible downtime reasons (empty areas in the visualization), we notice that rovers are often idle during communication actions done by other rovers. From the domain, we know that communications require a lander, which is occupied for the duration of the action. If set to highlight all actions that use the only available lander, *general*, the visualization shows that this lander is constantly engaged in communications after the 10-second mark, and is likely a bottleneck for optimization. Changing the problem file to add a second lander and generating a new plan would show a reduction in total time from 137 to 93 seconds.

Planner Debugging In this example, shown in Figure 7, we discuss how the visualization tool can be used for planner debugging. The plan was generated from a problem using in *office robot* domain from (Canal et al. 2022) and thought to be optimized for time. A quick way to look for issues in domain/problem modeling is to display causal links in all operators, one by one. Actions that have more or fewer links than they should are quickly recognizable. In this example, however, a different issue appears: one robot has two actions with no colored outline, i.e., their progress towards goals was either 0 or negative, and one of them undoes a goal condition. Zooming in and looking into action details, we can see that the robot is moving in circles, in and out of a waypoint. This behavior prompts an investigation into the planner, which is then found to have been executed with incorrect parameters and not properly optimizing the plan.

User Evaluation

We conducted two studies to test our model, both coupling a formal experiment measuring performance across visual representations with open ended questions to elicit subjective preference and insight into differences in how visual encoding of information may be read and interpreted. The first was a pilot study, conducted with 9 participants, from which 4 were male and 5 were female; 4 declared having knowledge in AI Planning, while 5 had no particular knowledge but a computer science background. 5 of them were in between ages 20 and 29, 3 between 30 and 39 and 1 over age 40. The second was a comparative study, with 39 participants, from which 24 were male, 13 were female, 2 were non-binary; 18 declared basic knowledge in AI Planning or a computer science background, 11 declared no knowledge in AI Planning, 9 declared some knowledge, 1 declared expert level knowledge; 27 were aged between 18 and 22, 10 were aged between 22 and 26, 2 were aged over 26.

Pilot study In the pilot study, participants were asked to answer questions related to six scenarios, each addressing one or more of **T1-T4**. Stimuli consisted of either a simplified PDDL-like plan description accompanied by its corresponding visual representation (Scenarios 1-2), or solely of a plan visual representation (Scenarios 3-6). Scenarios 1 and 2 focused on Action and Goal representation, the stimuli consisting of a plan pseudo-code description accompanied by its corresponding visual summary. Participants where asked to indicate at which time step an action had taken place or a goal had been completed. Scenarios 3 to 5 focused on visual representation alone, and plan description was omitted from the stimuli: Scenario 3 focused on interpretation of causal links, asking participants which of a set of objects was picked up first; Scenario 4 focused on interpretation of groups, asking participants to identify the number of instances of a given action in the plan; Scenario 5 focused on interpretation and concept grasping, asking participants to first match a pseudo-code plan to one of three visual summaries and then report differences between three visual summaries; Scenario 6 was optional and aimed at gauging participant engagement as well as their interpretation of a more complex visual representation integrating all elements tested

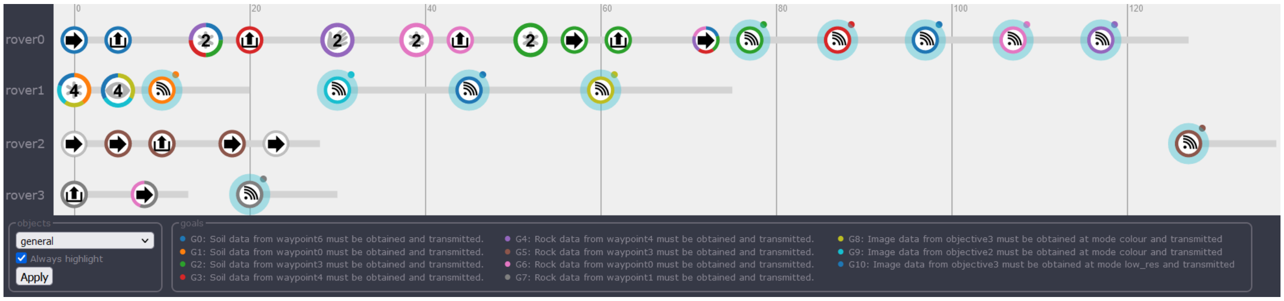

Figure 6: Plan for a IPC2002 *Rovers* domain problem. Four rovers perform several tasks capturing samples and images, optimized for time. All goals require obtained data to be transmitted to a lander, which is occupied for the duration of the transmission. By setting the visualization to highlight in blue all actions that involve the lander object *General*, we can observe that it is being used during most of the time and having only one lander is likely a bottleneck to this system.

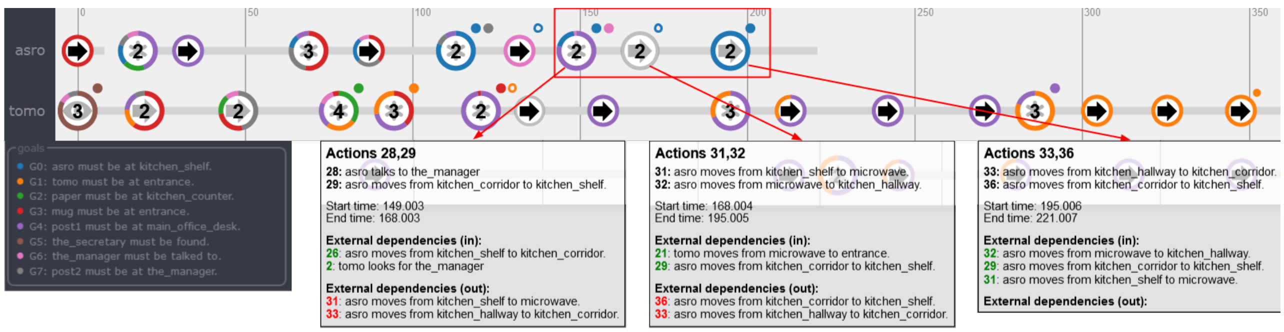

Figure 7: Plan for a *office robot* domain (Canal et al. 2022) problem. Two robots are performing tasks in an office. The area highlighted in red contains two actions that show no progress towards any goals while undoing one of them. This seems odd, so the user investigates further. Action descriptions show that Asro moves in a circle, leaving the kitchen shelf and returning to it afterwards for no apparent reason, indicating that the plan may have been poorly optimized.

in previous scenarios, asking participants to identify which action group included a given action.

For each scenario, participants were also asked: i) to rate the influence of the visual representation on their plan understanding and task completion, ii) to provide a brief description of what they thought was happening in the plan. For (i) a five-point Likert scale was used, with 1 being a strong negative impact and 5 a strong positive impact. (ii) was intended to gauge their understanding of plan and visual representation as a whole.

Performance results were promising, with tasks related to Scenarios 1, 3 and 5 reporting 100% accuracy, Scenario 2 achieving 89% accuracy (M=0.89, SD=0.33, SE=0.11), Scenario 6 achieving 63% accuracy (M=0.62, SD=0.52, SE=0.18) and Scenario 4 achieving 56% accuracy (M=0.56, SD=0.53, SE=0.18). Descriptive statistical results for Scenarios 2, 4 and 6 are reported in Table 1. Participants' descriptive responses all provided a consistent description of the actions in the plan for all scenarios, as well as correctly describing differences across plans. When rating the overall visualization as a complement to plans, the mean response was M=3.88. At the end, users were also asked to rate the difficulty of learning the visual representations from 1 (very easy) to 5 (very difficult). The mean response was M=3.25.

	Scenario 2	Scenario 4	Scenario 6
Mean	0.89	0.56	0.62
SD	0.33	0.53	0.52
SE	0.11	0.18	0.18
CI	[0.63,1.15]	[0.15, 0.96]	[0.19, 1.06]

Table 1: Mean, Standard Deviation, Standard Error and Confidence Intervals for pilot study scenarios 2, 4, 6.

Comparative Study The second study was designed to further compare our model to a baseline. We chose an approach similar to (Le Bras et al. 2020), comparing the proposed method against a Gantt chart. This study was divided into two parts. In the first, we devised four scenarios: Scenarios 1-2 contained written plans and Gantt charts for their actions, colored according to actor. Scenarios 3-4 contained written plans and a visualization using our proposed model. All plans were of similar complexity (3 goals, 16-20 actions, and 2 actors). Users were told that all plans referred to the same domain, robots in an office to which a floor plan was provided. All scenarios were shown as a questionnaire with static images, and the order in which they appeared was randomized. For all scenarios, participants had three tasks: (i) identify the order in which goals were achieved, (ii) answer a question regarding a parameter in a plan (which actor performed a given action, which object was involved in an action), (iii) provide a summary of what they thought happened in the plan. Participants did not have any previous contact with the visualization tool; an introduction page described basic aspects of the glyph visualization, and images in scenarios 3-4 contained a reference for the meaning of visual elements in a glyph. As the study was conducted with static images, glyphs were numbered as to reference which actions they represented. We also recorded how much time users spent on each scenario, as a way to obtain insights into the glyph system's learning curve. However, we did not enforce any time constraints on the participants. Participants were then asked to rate the usefulness of both the Gantt chart and the visualization from 1 (not effective at all) to 5 (extremely effective), choosing which method they perceived to be more useful, and to provide optional general feedback.

Table 2 shows the results for scenarios 1-4, with accuracy and standard deviation in the ordering task (i), and the question task (ii). When asked the order in which goals were achieved, participants attained a combined accuracy of 75.64% (SD=0.41, SE=0.07) for scenarios 1-2 and 79.48% (SD=0.40, SE=0.07) for scenarios 3-4. Time spent on scenarios 3-4 was on average 12% greater than on scenarios 1-2. The mean usefulness score for the Gantt chart was 2.44 (SD=1.03, SE=0.16), while the proposed model got a mean of 3.38 (SD=1.25, SE=0.20). When asked which method was the most useful, 69.23% chose the proposed visualization, 15.38% chose the Gantt chart, and 15.38% considered both to be equally effective/ineffective.

After scenarios 1-4 were complete, participants were exposed to the second part of the study: scenarios 5-6, which contained a matching task similar to scenario 5 in the pilot study: participants received a written plan and various different visualizations, and were tasked with identifying which of them corresponded to the written plan. Participants were also given an option to select if they were not sure. Alternative choices were designed as to provide identifying features in different visual channels, such as goals completed in different orders, actions undertaken by different actors, or different number of actors. Table 3 shows accuracy results from the plan matching tests. 69.23% of participants picked the correct representation in scenario 5, and 76.92% picked the correct representation on scenario 6. When compared in a one-sided t-test against a null hypothesis h_0 of choices being picked at random, both scenarios returned p-values below 10^{-7}, showing that there is likely a high level of correlation between image and plan for most users.

	Sc 1	Sc 2	Sc 3	Sc 4
Ordering Acc	0.69	0.82	0.77	0.82
Ordering SD	0.45	0.38	0.42	0.38
Ordering SE	0.07	0.06	0.07	0.06
Question Acc	0.92	1.00	0.97	0.95
Question SD	0.27	0.00	0.16	0.22
Question SE	0.04	0.00	0.03	0.04
Time Mean	215.80s	209.03s	246.88s	232.87s
Time SD	150.50s	148.87s	218.77s	138.08s
Time SE	24.10	23.84	35.03	37.29

Table 2: Results for comparative study, scenarios 1-4, per task (ordering and question) and mean total time spent in each scenario, incl. Standard Deviation and Standard Error.

	Sc 5	Sc 6
Selected correct plan	69.23%	76.92%
Selected wrong plan	10.25%	12.82%
Selected "not sure"	20.51%	10.26%
p-value against random	$3.3 * 10^{-8}$	$1.3 * 10^{-9}$

Table 3: Results for comparative study, scenarios 5-6, and p-value against h_0 (random choice) of one-sided t test.

Conclusion

In this paper, we presented a visualization method for AI Planning focused on observing different actors through a timeline, describing how it summarizes plan structure and displays goal status. We presented example applications and user feedback, as means of demonstrating its applicability.

Limitations Although we discuss goal progress as a temporal factor due to the timeline nature of the visualization, it is not related to time as an optimization constraint. Even if a plan is not optimized for time, observing temporal goal progress provides similar insights. Adapting Algorithm 1 to use a different variable can, however, be done without much additional effort. One limitation is that our approach requires the domain to explicitly represent actors; many domains include actions representing organizational processes or joint efforts from different parts, and the option would be to represent the entire system as a single actor. Finally, plans are assumed to be calculated beforehand; it would be interesting to investigate the overhead incurred by interactively generating new plans and providing visualizations on the fly.

Future Work Many research directions present themselves for future work. The taxonomy concepts presented can be further expanded as to perform statistical analysis on several application domains and develop a general hierarchical taxonomy for robotics that could be shared by different visualization approaches; A taxonomy-based glyph system for PDDL objects can be developed, allowing for the complete representation of an action and all of its parameters; Finally, a fully-customizable spatial layout system would also be a promising direction.

Acknowledgements

This work has been supported by the EPSRC grant THuMP (EP/R033722/1). Gerard Canal is supported by the Royal Academy of Engineering and the Office of the Chief Science Adviser for National Security under the UK Intelligence Community Postdoctoral Research Fellowship programme. The authors would like to thank Prof. Luc Moreau and Dr. Andrew Coles for insights and fruitful discussions.

References

Benton, J.; Smith, D.; Kaneshige, J.; Keely, L.; and Stucky, T. 2018. CHAP-E: A plan execution assistant for pilots. In *ICAPS*.

Borgo, R.; Cashmore, M.; and Magazzeni, D. 2018. Towards providing explanations for AI planner decisions. *arXiv preprint arXiv:1810.06338*.

Borgo, R.; Kehrer, J.; Chung, D. H.; Maguire, E.; Laramee, R. S.; Hauser, H.; Ward, M.; and Chen, M. 2013. Glyph-based Visualization: Foundations, Design Guidelines, Techniques and Applications. In *Eurographics (State of the Art Reports)*, 39–63.

Canal, G.; Krivić, S.; Luff, P.; and Coles, A. 2022. PlanVerb: Domain-Independent Verbalization and Summary of Task Plans. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36.

Cashmore, M.; Fox, M.; Long, D.; Magazzeni, D.; Ridder, B.; Carrera, A.; Palomeras, N.; Hurtos, N.; and Carreras, M. 2015. Rosplan: Planning in the robot operating system. In *Proceedings of the International Conference on Automated Planning and Scheduling*, volume 25.

Chakraborti, T.; Fadnis, K. P.; Talamadupula, K.; Dholakia, M.; Srivastava, B.; Kephart, J. O.; and Bellamy, R. K. 2017. Visualizations for an explainable planning agent. *arXiv preprint arXiv:1709.04517*.

Chakraborti, T.; Sreedharan, S.; Grover, S.; and Kambhampati, S. 2019. Plan Explanations as Model Reconciliation–An Empirical Study. In *2019 14th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, 258–266.

Choo, J.; and Liu, S. 2018. Visual analytics for explainable deep learning. *IEEE computer graphics and applications*, 38(4): 84–92.

Collins, A.; Magazzeni, D.; and Parsons, S. 2019. Towards an argumentation-based approach to explainable planning. In *ICAPS 2019 Workshop XAIP Program Chairs*.

Dhurandhar, A.; Chen, P.-Y.; Luss, R.; Tu, C.-C.; Ting, P.; Shanmugam, K.; and Das, P. 2018. Explanations based on the missing: Towards contrastive explanations with pertinent negatives. *arXiv preprint arXiv:1802.07623*.

Fox, M.; Long, D.; and Magazzeni, D. 2017. Explainable Planning. In *IJCAI-17 Workshop on Explainable Planning*.

Hayes, B.; and Shah, J. A. 2017. Improving robot controller transparency through autonomous policy explanation. In *International Conference on Human-Robot Interaction*.

Hoffmann, J.; and Magazzeni, D. 2019. Explainable AI planning (XAIP): overview and the case of contrastive explanation. *Reasoning Web. Explainable Artificial Intelligence*, 277–282.

Kumar, A.; Vasileiou, S. L.; Bancilhon, M.; Ottley, A.; and Yeoh, W. 2021. VizXP: A Visualization Framework for Conveying Explanations to Users in Model Reconciliation Problems. In *ICAPS 2021 Workshop on Explainable AI Planning*.

Le Bras, P.; Carreno, Y.; Lindsay, A.; Petrick, R. P.; and Chantler, M. J. 2020. PlanCurves: An Interface for End-Users to Visualise Multi-Agent Temporal Plans. In *ICAPS Workshop on KEPS*.

Lindsay, A. 2020. Using generic subproblems for understanding and answering queries in XAIP. *KEPS*.

Liu, S.; Wang, X.; Liu, M.; and Zhu, J. 2017. Towards better analysis of machine learning models: A visual analytics perspective. *Visual Informatics*, 1(1): 48–56.

Long, D.; and Fox, M. 2003. The 3rd international planning competition: Results and analysis. *Journal of Artificial Intelligence Research*, 20: 1–59.

Madumal, P.; Miller, T.; Vetere, F.; and Sonenberg, L. 2018. Towards a grounded dialog model for explainable artificial intelligence. *arXiv preprint arXiv:1806.08055*.

Magnaguagno, M. C.; Pereira, R. F.; and Meneguzzi, F. 2016. DOVETAIL—An Abstraction for Classical Planning Using a Visual Metaphor. In *Intl. Flairs Conference*.

Maguire, E.; Rocca-Serra, P.; Sansone, S.-A.; Davies, J.; and Chen, M. 2012. Taxonomy-based glyph design—with a case study on visualizing workflows of biological experiments. *IEEE Transactions on Visualization and Computer Graphics*, 18(12): 2603–2612.

Mcdougall, S. J.; Curry, M. B.; and De Bruijn, O. 1999. Measuring symbol and icon characteristics: Norms for concreteness, complexity, meaningfulness, familiarity, and semantic distance for 239 symbols. *Behavior Research Methods, Instruments, & Computers*, 31(3): 487–519.

Miller, T. 2018. Contrastive explanation: A structural-model approach. *arXiv preprint arXiv:1811.03163*.

Miller, T. 2019. Explanation in artificial intelligence: Insights from the social sciences. *Artificial intelligence*, 267.

Rees, D.; Laramee, R. S.; Brookes, P.; D'Cruze, T.; Smith, G. A.; and Miah, A. 2020. AgentVis: Visual Analysis of Agent Behavior withHierarchical Glyphs. *IEEE transactions on visualization and computer graphics*.

Rosenthal, S.; Selvaraj, S. P.; and Veloso, M. M. 2016. Verbalization: Narration of Autonomous Robot Experience. In *IJCAI*, 862–868.

Seegebarth, B.; Müller, F.; Schattenberg, B.; and Biundo, S. 2012. Making hybrid plans more clear to human users-a formal approach for generating sound explanations. In *International Conference on Automated Planning and Scheduling*.

Sreedharan, S.; Chakraborti, T.; Muise, C.; Khazaeni, Y.; and Kambhampati, S. 2020. –D3WA+–A Case Study of XAIP in a Model Acquisition Task for Dialogue Planning. In *Proceedings of the International Conference on Automated Planning and Scheduling*, volume 30, 488–497.

Sreedharan, S.; Olmo, A.; Mishra, A. P.; and Kambhampati, S. 2019. Model-free model reconciliation. *arXiv preprint arXiv:1903.07198*.

Sreedharan, S.; Srivastava, S.; and Kambhampati, S. 2018. Hierarchical Expertise Level Modeling for User Specific Contrastive Explanations. In *IJCAI*, 4829–4836.

Tintarev, N.; Kutlak, R.; Masthoff, J.; Van Deemter, K.; Oren, N.; and Vasconcelos, W. W. 2014. Adaptive Visualization of Plans. In *UMAP Workshops*.