Accepted to the 30th International Working Conference on Requirement Engineering: Foundation for Software Quality (REFSQ).

What Impact do my Preferences Have?

A Framework for Explanation-Based Elicitation of Quality Objectives for Robotic Mission Planning

Rebekka Wohlrab[1][0000−0002−5449−7900], Michael Vierhauser[2][0000−0003−2672−9230], and Erik Nilsson[1]

[1] Chalmers | University of Gothenburg, Gothenburg, Sweden
wohlrab@chalmers.se; erik1.nilsson2@gmail.com
[2] University of Innsbruck, Innsbruck, Austria
Michael.Vierhauser@uibk.ac.at

Abstract. *[Context and motivation]* Successful human-robot collaboration requires that humans can express their requirements and that they comprehend the decisions that robots make. Requirements in this context are often related to potentially conflicting quality objectives, such as performance, security, or safety. Humans tend to have preferences regarding how important different objectives are at different points in time. *[Question/problem]* Currently, preferences are often expressed based on assumptions of what importance level should be assigned to a quality objective at runtime. To assign meaningful preferences to quality objectives, it is important that humans understand the impact of these preferences on the behavior of a robot. To the best of our knowledge, there is yet no framework that supports the explanation-based elicitation of quality preferences. *[Principal ideas/results]* To address these needs, we have developed OBJUST, a framework that helps with the interactive elicitation of preferences for robot mission planning. *[Contribution]* The framework relies on the specification of human preferences and contrastive explanations. We evaluated our framework in a study with 7 participants. Our results indicate that the visual and textual explanations of the generated robotic mission plans help humans better understand the impact of their preferences, which can facilitate the elicitation process.

Keywords: quality attributes · elicitation · robot mission planning · contrastive explanation

1 Introduction

To enable human-robot collaboration, humans need to be able to *express* their (potentially changing) objectives. To support the elicitation of objectives, it is crucial that humans *understand* `how` robots work, `what` tasks they are performing, and `why` robots select particular actions that were considered optimal in a given situation [23]. In practice, multiple objectives are used in robot mission planning, many of which are quality attributes such as performance, energy consumption, safety, or security [35,34,28].

In certain situations, a quality objective might become more important (e.g., due to an arising safety hazard, where a robot should avoid a location), resulting in the need to replan missions [11]. However, it is not always obvious how different requirements result in different plans [45]. Previous work found that humans frequently struggle with understanding how their preferences of different quality objectives affect the automated planning process [2]. The need for explanations as a guiding tool in requirements engineering for self-adaptive systems has been raised [41]. To the best of our knowledge, there is no human-on-the-loop approach that helps humans to elicit changing preferences supported by explanations [49,29]. It is not enough to elicit input once at the beginning, but the dynamic nature of run-time contexts and stakeholder preferences [41,49] may require recalculating a mission with new or updated references.

Stakeholders working with robots are not always people directly interacting with a robot, but may be supervisors who operate systems or observe them at a distance. For example, in a warehouse, there might be few humans who directly communicate with a robot. For these scenarios, it is beneficial to have a user interface that can help humans get an overview of the planning problem, indicate quality objectives, and understand the tradeoffs of a particular plan [19,13].

In this paper, we present the OBJUST framework that enables humans to *express their preferences* for robot mission planning, *provides explanations* of plans that come with different tradeoffs, and further helps humans *adjust preferences* if necessary. While many existing approaches rely on textual explanations only [45], our approach supports a combination of textual and visual explanations. OBJUST does not propose or prescribe a specific requirements elicitation technique, but provides a framework to help stakeholders assess the effect of different preferences on the expected outcome of a planning problem. In a preparatory step, requirements are collected from stakeholders, e.g., using brainstorming, interviews, or workshops [10,16,4]. The problem is that these requirements are often elicited based on assumptions about how systems will act at runtime and how preferences might impact the behavior of the final system. Making such assumptions is not feasible and desirable in practice. To mitigate this issue, once an initial set of requirements has been collected, OBJUST can be used to investigate the effect of different preferences on the planning results. We put a special emphasis on providing a comprehensive explanation of why certain paths were deemed optimal by the planning algorithm.

We claim three main contributions with our OBJUST framework. First, we provide a list of requirements for an elicitation and explanation framework. Second, we propose an architecture of our elicitation and explanation framework along with a domain model for the underlying knowledge base. To demonstrate the framework's applicability, we provide an open-source prototype implementation. Finally, we provide a user dashboard that supports user input and explainability.

Sect. 2 describes the background and related work. Sect. 3 presents our research method. In Sect. 4 we then present OBJUST. We describe our evaluation in Sect. 5. Sect. 6 presents the threats to validity. In Sect. 7, we discuss the results, limitations, and avenues for future research.

2 Background

Robotic systems need to consider different quality requirements, to ensure that the system can respond to changing conditions in the environment, such as changes in workload, resource availability, or emerging security threats [32]. Quality objectives are often encoded in *cost functions* representing the costs associated with a certain action or sequence of actions. For instance, the cost might represent the distance traveled, energy expended, or risk encountered [17,48]. In related work, these cost functions are often combined in a weighted sum [18,20]. However, it is not always obvious how the weights of such a global cost function should be set. Stakeholders require tools and decision-making techniques to assist their prioritization of quality objectives and reach consensus [41].

2.1 Motivating Example

We represent the mission planning problem as a graph, consisting of locations and edges. Fig. 1 depicts a map containing locations and edges that a robot can choose from to reach its destination, where the number over each edge is the distance between locations. In this example, we use three quality objectives: *travel time*, *safety*, and *privacy*. Safety is measured in terms of collisions and privacy in terms of the expected number of privacy intrusions.

Traversing a normal edge yields a safety cost of 0, a partially occluded edge costs 1, and an occluded edge costs 2. Passing a public location yields a privacy cost of 0, a semi-private location costs 1, and a private location costs 2.

To calculate the optimal path, we define the cost function for a plan σ as a weighted sum:

$$c(\sigma) = w_{\text{tt}} \cdot c_{\text{tt}}(\phi_{\text{tt}}(\sigma)) + w_{\text{col}} \cdot c_{\text{col}}(\phi_{\text{col}}(\sigma)) + w_{\text{int}} \cdot c_{\text{int}}(\phi_{\text{int}}(\sigma))$$

where $w_{\text{tt}}, w_{\text{col}}, w_{\text{int}} \in \mathbb{R}^+$, and $w_{\text{tt}} + w_{\text{col}} + w_{\text{int}} = 1$.

c_* is the local cost function for each quality attribute, $\phi_*(\sigma)$ is the total cost of each attribute in a path, and w_* is the weight of each quality attribute. Weights are used to encode stakeholders' preferences concerning the importance of that quality attribute. The local cost function c_* is calculated by quantifying the

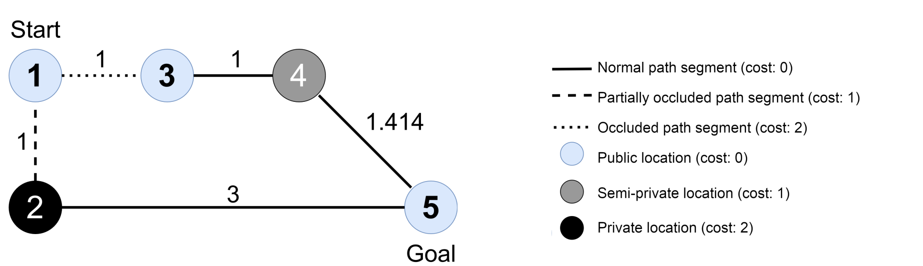

Fig. 1: A planning graph. The node colors represent 3 node types. The line styles (solid, dashed, dotted) represent normal, semi-occluded, and occluded edges.

Table 1: Cost function values of paths for different preferences.

w_{tt}	w_{col}	w_{int}	Locations	Optimal	ϕ_{tt}	ϕ_{col}	ϕ_{int}	c_{tt}	c_{col}	c_{int}	c
1	0	0	3,4	Yes	3.414	2	1	1	2	1	1
			2	No	4	1	2	1.172	1	2	1.172
0	1	0	3,4	No	3.414	2	1	1	2	1	2
			2	Yes	4	1	2	1.172	1	2	1
0.25	0.5	0.25	3,4	No	3.414	2	1	1	2	1	1.5
			2	Yes	4	1	2	1.172	1	2	1.293

local cost for a path in relation to the least expensive path's cost for the same attribute.

An automated planner may choose a path depending on the weighted preferences of quality objectives. Tab. 1 displays the attribute costs and cost function values for several sets of preferences. In the case where only travel time is relevant ($w_{tt} = 1$), the path 1 → 3 → 4 → 5 is optimal. If the system only cares about safety ($w_{col} = 1$), then the path 1 → 2 → 5 is deemed optimal.

Fully prioritizing a single quality attribute is easy, but mixing weights does not always yield intuitive paths. E.g., for the preferences $w_{tt} = 0.25, w_{col} = 0.5, w_{int} = 0.25$, the path 1 → 2 → 5 is deemed optimal by the robot. However, why exactly was this path chosen? Even in this small graph, it is not always easy to calculate this by hand, or by intuition. For large maps, identifying the optimal path requires error-prone and time-consuming calculations. Therefore, solutions are needed that help humans to understand the consequences of their preferences.

2.2 Related Work

Priority awareness: Samin et al. [39,40] have coined the concept of priority awareness with their Pri-AwaRE approach, to automatically adjust priorities to satisfy QoS requirements. Priorities are similar to preferences in OBJUST. Constraints can be expressed as well, but are not the focus of this paper. It appears promising to combine OBJUST with Pri-AwaRE, so that explanations can be given with respect to why certain requirements are fulfilled and what impact the adjusted priority values have.

Besides cost functions, there exist other ways of representing priorities of quality objectives. For example, some existing approaches select so-called knee solutions among a set of Pareto-optimal solutions [8,22]. OBJUST is similar in the sense that it selects Pareto-optimal solutions, takes the balanced points by default (if all priorities are the same, as in our example), but allows to deviate from them when quality objectives are reprioritized.

The need to dynamically adjust requirements for self-adaptive systems has been addressed, for example, by frameworks to enable goal model adaptation [25,34].

It focuses on run-time verification of adaptable goals. Similarly, approaches based on KAOS [1,9] have a general focus on functional goals and uncertainty. While goal modeling can capture quality-related aspects as well, our focus lies more on quality attributes and the impact that quality preferences have on generated plans. In this area, Bryl et al. [3] explored the use of goal modeling and requirements analysis, in conjunction with planning techniques using Tropos. Similar to our work, they describe a tool-supported approach for analyzing and exploring alternative requirements. Their work does not lie on the changing assignment of different weights to soft goals, whereas the assignment of preferences for different quality attributes is exactly the focus of OBJUST. Our previous work [52] has addressed the issue of run-time adaptation of quality attributes. We elaborated on the challenges and proposed steps towards quality attribute adaptation. Similarly, Li et al. developed a framework for preference adaptation and concluded that future works need to explain the impact of preferences to users [29]. This is the research gap we are addressing in this paper: making the impact of preferences explicit so that they can be better elicited.
Explainable planning: In recent years, the area of explainable planning has received increased attention [24]. Contrastive explanations are among the most common forms of explanation proposed by related work [14,27,37,7].They can help human users identify potential biases and errors in a robot's decision-making [33]. Contrastive explanations have been used to help humans understand why a robot mission plan was optimal for a given set of quality objectives [45]. Eifler et al. [14] developed a user interface that allows human users to iteratively explore the planning space, ask "why not" questions, and specify planning goals. Existing approaches often focus on in-situ explanations, in which a robot explains a current action it just took or is about to take [6,42]. OBJUST focuses on explanations of multiple planning alternatives for entire missions, to help humans give appropriate input during interactive robot mission planning.
Preference elicitation: While traditional explainability approaches for robot mission planning have focused on describing why one plan is optimal or better than another, we focus on giving input to users who specify their (potentially changing) preferences to facilitate replanning at runtime.
Shaikh et al. [43] proposed a related approach that relies on a GUI. Similar to OBJUST, they also use sliders to indicate the importance of different quality objectives. Besides sliders, also a palette interface and a prism interface were implemented. Both the slider and the palette interfaces were found to be usable. OBJUST extends the use of interfaces for preference elicitation by providing a visualization and explanation component that can help humans when interactively exploring how their preference selection impacts the generated plans. In that way, preferences can be adjusted so that the desirable behavior is achieved.

3 Research Method

We applied design science [50] with several iterations of solution design and validation. In this paper, we focus on the framework for interactive elicitation and

Table 2: Overview of the study participants, their occupations, and experiences.

Part.	Occupation	Experience with technical subjects
1	Engineering manager	3-5 yrs.
2	Graduating software development student	1-2 yrs.
3	Graduating software engineering student	1-2 yrs.
4	Software developer	6+ yrs.
5	Backend developer	3-5 yrs.
6	Cloud engineer/architect	6+ yrs.
7	UX-design student	0 yrs.
8	Consultant manager	6+ yrs.
9	Software architect	6+ yrs.
10	Product owner	6+ yrs.

explanation as the design artifact. The goal of the research was to better *understand humans' needs* for explanation to guide the interactive elicitation of quality preferences, to *develop* the OBJUST framework supported by a prototype, and to *evaluate* to what extent the framework fulfills the needs of our participants. Our process consisted of (0) *identifying shortcomings* of existing solutions, (1) a *requirements elicitation phase* with human participants; (2) the *development* of the conceptual domain model and prototype implementation; and (3) an *evaluation phase* of the prototype. Interview material can be found on Figshare[3], and the implementation of our prototype is available online[4].

Table 2 shows an overview of the participants. All participants had experience with UX or technical subjects, with levels of experience ranging from senior students to practitioners who had worked in industry for more than 6 years.

Participation was voluntary, the participants were asked to give consent to participate in the study, and the procedures were explained. All participants were informed about their anonymity and assured that they could withdraw from the study at any point in time. No personally identifiable information was collected. The conducting researcher took notes and recorded the data from the survey.

After the interviews, the data was coded [15] using the QualCoder[5] tool for the thematic analysis. Codes were created in an iterative way and structured into categories of codes as a tree to arrive at our findings.

Requirements Elicitation Phase: We performed interviews with Part. 1–7 to investigate challenges when expressing preferences and understanding plans. Part. 8–10 were not available but participated in the evaluation phase instead. One author performed the elicitation and was present in all sessions. The sessions relied on pen-and-paper calculations, in which the participants were asked what path would be optimal for a given set of quality objective weights. We encouraged the participants to think aloud. The participants were also asked questions about

[3] `https://doi.org/10.6084/m9.figshare.24006978.v1`
[4] `https://github.com/SE-CPS/OBJUST_public`
[5] `https://qualcoder.wordpress.com`

what features would be useful to alleviate such a task. The questions can be found in the supplementary material[3] and were mainly concerned with the participants' reasoning and perceived difficulties. Furthermore, participants were also asked for suggestions of features that might help to mitigate their challenges.
The data from the elicitation phase was then coded by the researcher who performed the interviews. Afterwards, the codes were grouped into themes and discussed in a data analysis workshop. We found that the challenges from the participants can be addressed by a number of requirements. As an outcome of this phase, we collected a set of six requirements/core features (cf. Sect. 4.1) serving as the input for the prototype implementation.
Development: We developed the framework based on the data from the elicitation phase. We systematically went through the requirements and understood what features were needed in a prototype that addressed the participants' challenges. We also developed a domain model by understanding the key concepts that were needed to reason about the robot mission planning domain.
Think-Aloud Study for Evaluation: Part. 1, 2, 5, and 7–10 were involved in this phase. The other participants were asked to participate but were not available. In the evaluation phase, participants worked on different tasks with the tool and were asked to fill out a short survey.

4 Framework for the Interactive Preference Elicitation

In this section, we introduce the elicited requirements for our framework (Sect. 4.1), the core implementation (Sect. 4.2), and visualization and explanation features (Sect. 4.3).

4.1 Requirements for Interactive Elicitation and Explanation

Based on our analysis, we arrived at a list of the following six functional requirements for an explanation visualization framework:

(R1) *The system shall allow the user to prioritize quality objectives:* R1 is concerned with the importance of eliciting the relevance of each quality objective, which in turn serves as an important input to automated planning.

(R2) *The system shall display an optimal path on a map or graph, along with its costs:* In the elicitation phase, we found that identifying the optimal path took a lot of time and it was difficult for participants to manage the complexity of planning problems quickly.

(R3) *The system shall display alternative paths with their costs:* R3 is relevant to compare a selected path to one that might seem optimal, but is not. This visual contrastive explanation was considered beneficial by the participants.

(R4) *The system shall provide a textual explanation of why a path is optimal:* During the elicitation, we found that participants struggled with the required calculations to understand what route was optimal given a cost function.

(R5) *The system shall indicate important nodes that distinguish one path from other paths:* Indicating differences between paths that are connected to key decision points in the map was considered crucial by the participants.

(R6) *The system shall support traceability between quality (input) data and generated plans:* R6 ensures that different paths can be easily traced back to the corresponding input requirements.

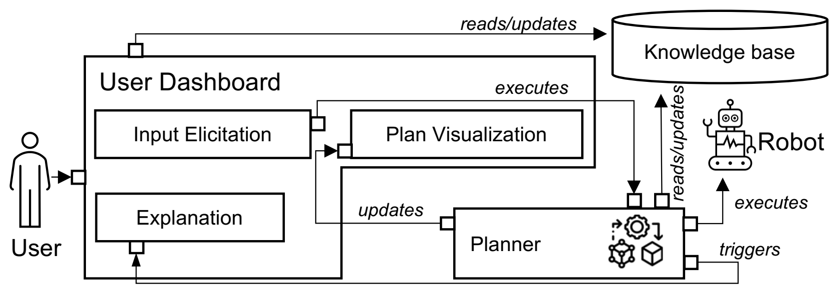

Fig. 2: Overview of the elicitation and explanation framework

4.2 Core Implementation

Fig. 2 provides an overview of the framework, which is described in the following. **Knowledge base:** Since it is often useful to explicitly capture concepts and relevant elements in a domain model [31], we have created a model for the adaptive planning of quality objectives. It supports traceability between quality data and the generated plans (R6). The model is used as the basis for the knowledge base and instantiated for each plan generation. For the initial instantiation, we reuse graph data from a public repository[6].

Fig. 3 provides an overview of the three main parts of the domain model. First, the *Qualities* part (top) describes the quality input provided by stakeholders, i.e., their constraints and preferences. Stakeholders can indicate how important a quality objective is and define constraints, for example, to restrict the value of a specific quality objective measure to a certain range. The quality objectives and preferences are then combined in a single *Cost Function*, e.g., as a weighted sum. Second, the *Environment* part (bottom) captures the structure of the *Map*, i.e., *Edges* (that may be occluded) and *Locations* with their privacy levels. For edges, a probability of an edge being successfully traversed can be specified. Finally, the *Planning Output* part (middle) represents the result of the automated planning. The Qualities part and the Environment part are input by humans, or given by the planning context. The Planning Output part (i.e., the set of locations that shall be visited) is fed into the robot for plan execution.

[6] https://github.com/cmu-able/explainable-planning

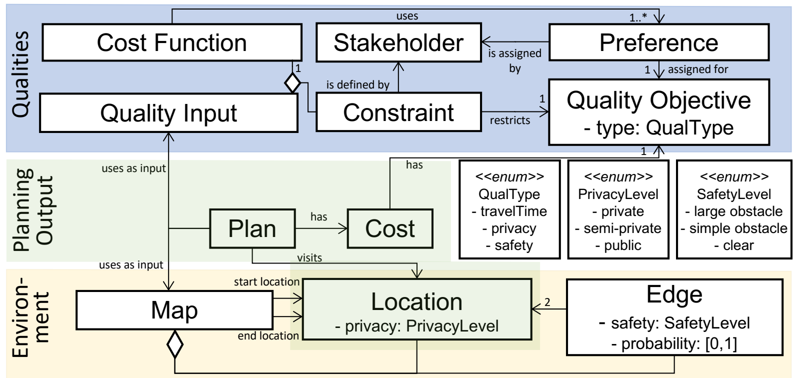

Fig. 3: Domain model for the underlying knowledge base

Input Elicitation: Fig. 4 shows an overview the user dashboard. The form on the left is used to elicit the users' preferences 1 for quality objectives (R1). For the input elicitation, the map, start and end locations, and preferences are specified. For the specification of preferences, the Analytic Hierarchy Process [38] is used. Users can use sliders to perform pairwise comparisons between quality objectives and indicate how much they prefer one quality objective over another. Based on this input, values between 0 and 1 can be computed to set the weights of different objectives. This approach has been applied to elicit preferences for different objectives before [51,30] and we considered it useful for this framework as well.

Planner: In our implementation, we opted for Dijkstra's path-planning algorithm [12,47]. Particularly for robotic applications, a wide variety of mission planning and path planning algorithms have been proposed [26,46]. OBJUST provides a flexible component-based framework that allows to easily exchange the planning component and use a different algorithm. The only requirement is for the algorithm to support multi-objective optimization, and that it can output an optimal path, along with the resulting costs of different quality objectives. For instance, A* [21] can be considered, as it is superior in time efficiency compared to Dijkstra [44,5]. For the heuristics, travel time can be approximated by using the Manhattan distance. For privacy or safety, it is not obvious what the admissible heuristics should be.

All quality objectives in the cost function are normalized to ensure that an objective with a generally higher cost cannot dominate others. We normalize objectives by comparing the cost for each quality objective in the current path to the lowest possible cost of that objective. Normalizing against the minimal cost instead of the maximum cost was deemed optimal because it requires less computational power ($\mathcal{O}(n \log n)$), compared to using, e.g., depth-first search ($\mathcal{O}(n^2)$) to find the maximum cost.

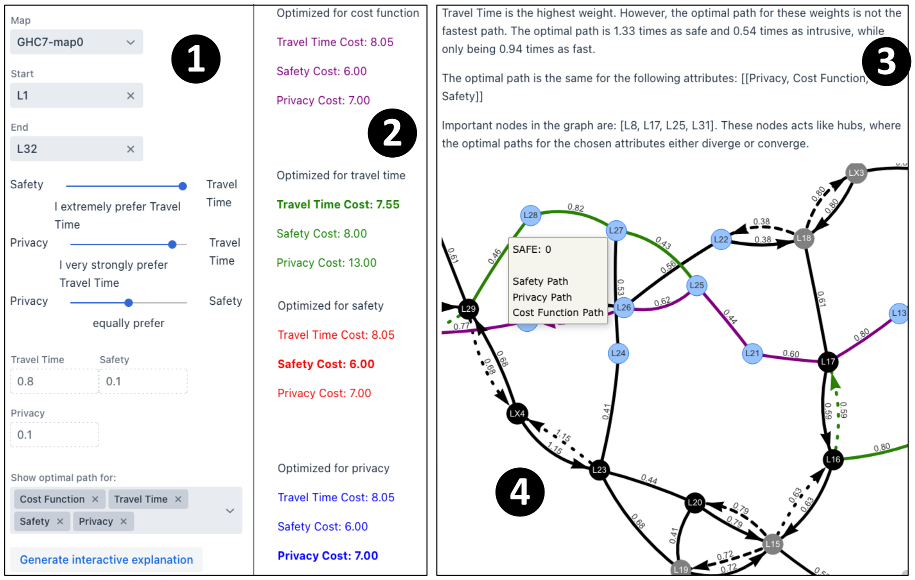

Fig. 4: Screenshot of the user dashboard, with input elicitation (1), cost overview (2), textual explanation (3), and plan visualization (4)

4.3 Visualization and Explanation

The framework *vis-network*[7] was chosen to display the graphs. An example of the visualization can be seen in Fig. 4 4.
All participants in the elicitation phase requested a tool to display multiple optimal paths at once since it facilitates a comparative analysis of alternative paths. When multiple paths overlap, it can become a visual clutter. Tooltips are useful in distinguishing overlapping paths. The mouseover tooltip feature that can be seen in Fig. 4 indicates the paths that are traversing a specific edge in the graph, as well as the properties of different edges or locations (R2). In the example, it can be seen that both the safety path, the privacy path, and the path optimizing for the cost function traverse the selected edge, which has a safety cost of 0. The tooltip provides an instant overview, removing the need for a separate legend for cost details.
Displaying the optimal path was considered the most crucial feature in the elicitation phase. The interviewees did not only state that they would like the framework to visualize the selected path, but also the optimal paths if you optimized for only one quality objective. Therefore, the tool indicates both the plan optimized for the global cost function (where the quality objectives' weights from the input elicitation are factored in), as well as the plans optimized for each individual quality objective. The various alternative paths are highlighted with

[7] https://visjs.github.io/vis-network/docs

different colors, so that users can easily distinguish what objectives a path was optimized for (R3). Examples can be seen in Fig. 4 in 2, 3, and 4.

Table with cost overview: One feature indicates the costs of different paths in a table-like structure 2. In the elicitation phase, several participants (Interviewees 1, 2, and 5) asked for a detailed table where the entries would contain different weights and the corresponding paths. The table color-codes the costs associated with each optimal path for easier identification. In Fig. 4, a few cost items in the list are highlighted in bold, implying that they represent the lowest cost for a specific quality objective. This feature aims to mainly combat the complexity that interviewees perceived during their tasks in the elicitation phase, removing the need to manually calculate each path's cost.

Textual explanation: OBJUST provides sentence-based feedback to clarify visual information 3 (R4). It is achieved using three features:

(i) Descriptive Text, clarifying why a specific path was chosen. In Fig. 4, it explains that even though *travel time* had the highest weight, the algorithm chose to optimize a path for other quality objectives because the difference in cost for safety and privacy was larger than the difference in *travel time*.

(ii) Equivalent Paths: This feature shows which objectives have the same costs for a path. It saves the users time when analyzing paths, especially in larger graphs.

(iii) Important Nodes (R5): This feature lists important nodes in the graph. These nodes act like hubs, where the optimal paths for the chosen objectives diverge or converge. The feature helps to reduce the complexity of larger graphs by segmenting them, enabling users to focus on a smaller portion of the graph.

5 Findings from our Think-Aloud Sessions

The goal of our think-aloud sessions was to investigate to what extent OBJUST fulfills the needs of our participants, with a particular focus on the explanation capabilities and interactive dashboard.

We conducted think-aloud sessions with 7 participants, performed by the same researcher as in the elicitation phase. We worked with a smaller map (6 locations) and then a larger one (37 locations). The researcher asked the participants to explain what path they would estimate to be optimal for different combinations of weights for quality objectives. Subsequently, the participants were asked to utilize the tool to solve the tasks. The participants also ranked each feature's usefulness.

Regarding the perceived usefulness of the prototype, all participants strongly agreed that *"The tool allows me to accomplish my tasks"*. The participants were satisfied with the tool and considered it to save them time. The most useful features were the explainability features. Even when dealing with small maps, the complexity of robot mission planning is so high that it is difficult for humans to identify an optimal path manually. No participant could provide an accurate or satisfactory explanation of why a specific path was chosen in the elicitation phase without the aid of the prototype, except if only a single quality objective was prioritized. With the tool, participants could give increasingly better explanations after each task.

Reducing the need for time-consuming calculations: Without OBJUST, participants had to calculate multiple paths and their costs manually to understand which path was optimal. The operations were mainly a combination of sums and products. Still, many participants struggled with the calculations and finding an optimal path quickly. When evaluating the prototype, both the text-based features and the visualization of paths were considered helpful. For example, Part. 2 stated that *"displaying multiple paths was really nice to quickly get an overview, instead of having to calculate them by hand"*.

The text features were perceived as particularly helpful by participants with limited knowledge of algorithms. To complement the text features, the visual features provided a quick and simple explanation of what the outcome was, removing the need for a separate legend. Part. 7 stated that *"it was quick and easy to find the optimal path, and I could then use the other features to understand why it was chosen"*. The tooltips indicating equivalent paths were considered beneficial as well, as they reduced the need to examine multiple individual paths and helped participants focus on groups of paths instead.

Reducing complexity: In the elicitation phase, we found that the complexity associated with finding the optimal path was challenging for our participants. Fig. 5 depicts that in the eyes of the participants, several features were helpful in reducing the perceived complexity of the planning problem. The participants deemed it crucial to display the optimal path in large graphs, as they couldn't see themselves calculating it by hand, no matter the time limit: *"There is no way I could ever find the optimal path in the large maps by myself, let alone multiple optimal paths."* (Part. 1).

The cost list and descriptive text were found useful in reducing the complexity: *"I found the cost list super useful because it allows me to easily compare the costs no matter how complex the paths are."* (Part. 8).

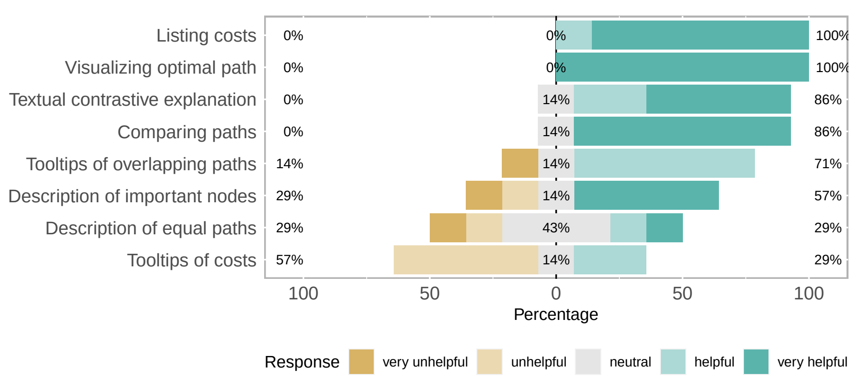

Fig. 5: Responses on features' *helpfulness in reducing the perceived complexity.*

Part. 9 liked that *"the graph gets larger, but the descriptive text explanation remains concise"*. When asked which features were their favorites, every single participant mentioned the cost list. Part. 7 stated that they liked the *"display of costs for each path. It is good with the color differentiation and clear language"* .

6 Threats to Validity

External Validity: The number of study participants was limited, which constitutes a threat to external validity. Further workshops and more participants are needed to strengthen the findings. With a larger number of participants, we will be able to perform experiments in a controlled setting and draw statistically significant conclusions about the usefulness and usability of OBJUST.
Another issue is that different levels of experience might influence how useful the participants think the framework is. To mitigate this, participants with a mix of experience were chosen. The external validity is also compromised by the focus on an example with a fixed set of quality objectives. Furthermore, we have only investigated the problem in one example domain, with a single implementation using one path planning algorithm (Dijkstra). To further confirm the applicability of our framework in a broader context, additional evaluation is necessary, including a comparison with other mission planning algorithms. However, the main focus of this paper was not the use of mission planning algorithms, but rather developing an approach to improve understandability and reduce the perceived complexity for humans.
Construct Validity: The quality attributes used in our prototype were easy to understand for all participants. However, the participants in our study might not have the same interpretation of words such as "quality objective", "plan", or "preference". This could have led to issues in our qualitative analysis. It was therefore crucial to spend a few minutes with each participant to establish a common terminology.
Internal Validity: Possible misunderstandings in the interviews and analysis may have led to incorrect conclusions. Asking participants to answer Likert-scale questions allowed for data triangulation with the data from the think-aloud sessions.
Reliability: The reliability of this study may be influenced by our interpretations. They may have affected the conclusions drawn from the data. To mitigate this threat, we aimed to clearly describe our methods and keep a transparent chain of evidence. All interview guides and questionnaire answers were made public[3].

7 Discussion and Future Work

We presented OBJUST, a framework for the interactive elicitation and explanation of quality-oriented mission planning. Our evaluation has indicated that the framework is useful for eliciting quality preferences, supported by explanations of their impact on generated plans. To the best of our knowledge, there is no approach that focuses on this gap and combines both visual and textual explanations.

In the following, we discuss three major findings:

Use of contrastive explanations: Notably helpful features were the concurrent display of optimal paths for different quality objectives, and their comparison using a list of their costs. Our findings confirm previous works about the usefulness of contrastive explanations [45,36].

Use of elicitation techniques: In previous works on elicitation, it was found that it is non-trivial for humans to understand the impact of a set of priorities on robot mission plans [51,41]. Therefore, OBJUST includes both an elicitation and an explanation component. We found that without a clear explanation, it is extremely difficult to understand what plan a given set of preferences leads to.

Visual and textual elements: Our participants stated that the visual features were useful to very quickly grasp the optimal path, compared to manually calculating it. Visualizations are useful for explaining *what* plan is deemed optimal. However, visual features are harder to generalize and require more development time compared to text-based features. To apply OBJUST to other systems, it would be necessary to design appropriate visualizations that are domain-specific. Text-based features are great at explaining *why* a plan was deemed optimal. They are also highly generalizable for different types of robotic systems. The domain model/vocabulary used in the textual explanation can be adjusted, so that it is easy to generate explanations for another domain and system.

The presented evaluation is only a pilot study. In the future, we plan to conduct a study involving more participants with both technical and non-technical backgrounds. We would like to involve more practitioners and preferably no students. Such an evaluation would help us to assess whether the approach is applicable in practice and how much the explanations help end users. We envision different versions of OBJUST, depending on the system at hand and the concrete setup. The GUI of OBJUST can be used for users to monitor the real-time behavior of a robot, intervene when necessary, and specify different preferences depending on what is desired in a given context. We expect that the general conceptual framework presented in this paper can be reused and then tailored to specific contexts and systems. For contexts with many quality objectives and many possible locations, mechanisms are needed to hide and display relevant information, so as not to overwhelm the user.

Acknowledgments: This work was partially supported by the Wallenberg AI, Autonomous Systems and Software Program (WASP) funded by the Knut and Alice Wallenberg Foundation.

References

1. Baresi, L., Pasquale, L., Spoletini, P.: Fuzzy goals for requirements-driven adaptation. In: Proc. of the 18th Int'l Requirements Eng. Conf. pp. 125–134 (2010)

2. Bowers, K.M., Fredericks, E.M., Cheng, B.H.: Automated optimization of weighted non-functional objectives in self-adaptive systems. In: Proc. of the 10th Int'l Symp. on Search-Based Soft. Eng. pp. 182–197. Springer (2018)
3. Bryl, V., Giorgini, P., Mylopoulos, J.: Supporting requirements analysis in tropos: a planning-based approach. In: Proc. of the Pacific Rim Int'l Conf. on Multi-Agents. pp. 243–254. Springer (2007)
4. Byrne, J.G., Barlow, T.: Structured brainstorming: A method for collecting user requirements. In: Proc. of the Human Factors and Ergonomics Society Annual Meeting. vol. 37, pp. 427–431. SAGE Publications Sage CA: Los Angeles, CA (1993)
5. Candra, A., Budiman, M.A., Hartanto, K.: Dijkstra's and A-star in finding the shortest path: A tutorial. In: Proc. of the 2020 Int'l Conf. on Data Science, Artificial Intelligence, and Business Analytics (DATABIA). pp. 28–32. IEEE (2020)
6. Chakraborti, T., Sreedharan, S., Grover, S., Kambhampati, S.: Plan explanations as model reconciliation. In: Proc. of the ACM/IEEE Int'l Conf. on Human-Robot Interaction (HRI). vol. 2019-March, pp. 258–266. IEEE (2019)
7. Chen, S., Boggess, K., Feng, L.: Towards transparent robotic planning via contrastive explanations. In: Proc. of the IEEE/RSJ Int on Intelligent Robots and Systems (IROS). pp. 6593–6598 (2020)
8. Chen, T., Li, K., Bahsoon, R., Yao, X.: FEMOSAA: Feature-guided and knee-driven multi-objective optimization for self-adaptive software. ACM Transactions Soft. Eng. Methodology **27**(2) (jun 2018)
9. Cheng, B.H.C., Sawyer, P., Bencomo, N., Whittle, J.: A goal-based modeling approach to develop requirements of an adaptive system with environmental uncertainty. In: Proc. of the Int'l Conf. on Model Driven Eng. Languages and Systems. pp. 468–483. Springer Berlin Heidelberg, Berlin, Heidelberg (2009)
10. Coulin, C., Zowghi, D., Sahraoui, A.E.K.: A situational method engineering approach to requirements elicitation workshops in the software development process. Software Process: Improvement and Practice **11**(5), 451–464 (2006)
11. Dennis Ding, X., Englot, B., Pinto, A., Speranzon, A., Surana, A.: Hierarchical Multi-objective planning: From mission specifications to contingency management. Proc. of the Int'l Conf. on Robotics and Automation pp. 3735–3742 (2014)
12. Dijkstra, E.W.: A note on two problems in connexion with graphs. Numerische mathematik **1**(1), 269–271 (1959)
13. Driewer, F., Sauer, M., Schilling, K.: Discussion of challenges for user interfaces in human-robot teams. In: Proc. of the 3rd European Conf. on Mobile Robots (2007)
14. Eifler, R., Brandao, M., Coles, A., Frank, J., Hoffmann, J.: Evaluating plan-property dependencies: A web-based platform and user study. In: Proc. of the Int'l Conf. on Automated Planning and Scheduling. vol. 32, pp. 687–691 (2022)
15. Flick, U.: The SAGE handbook of qualitative data collection. SAGE Publications Ltd (2017)
16. Franco, Á.J., Assar, S.: Leveraging creativity techniques in requirements elicitation: A literature review. Requirements Engineering Magazine **2016**(02) (2016)
17. Garcia, M.P., Montiel, O., Castillo, O., Sepulveda, R., Melin, P.: Path planning for autonomous mobile robot navigation with ant colony optimization and fuzzy cost function evaluation. Applied Soft Computing **9**(3), 1102–1110 (2009)
18. Gasparetto, A., Zanotto, V.: Optimal trajectory planning for industrial robots. Advances in Engineering Software **41**(4), 548–556 (2010)
19. Goodrich, M.A., Schultz, A.C., et al.: Human–robot interaction: a survey. Foundations and Trends in Human–Computer Interaction **1**(3), 203–275 (2008)

20. Gulati, S., Jhurani, C., Kuipers, B., Longoria, R.: A framework for planning comfortable and customizable motion of an assistive mobile robot. In: Proc. of the Int'l Conf. on Intelligent Robots and Systems. pp. 4253–4260. IEEE (2009)
21. Hart, P., Nilsson, N., Raphael, B.: A formal basis for the heuristic determination of minimum cost paths. IEEE Transactions on Systems Science and Cybernetics **4**(2), 100–107 (1968)
22. Hassan, S., Bencomo, N., Bahsoon, R.: Minimizing nasty surprises with better informed decision-making in self-adaptive systems. In: Proc. of the 10th Int'l Symp. on Soft. Eng. for Adaptive and Self-Managing Systems. pp. 134–145 (2015)
23. Hellström, T., Bensch, S.: Understandable robots-what, why, and how. Paladyn, Journal of Behavioral Robotics **9**(1), 110–123 (2018)
24. Hoffmann, J., Magazzeni, D.: Explainable AI planning (XAIP): Overview and the case of contrastive explanation. In: Lecture Notes in Computer Science. vol. 11810 LNCS, pp. 277–282 (2019)
25. Iftikhar, M.U., Weyns, D.: ActivFORMS: Active formal models for self-adaptation. In: Proc. of the 9th Int'l Symp. on Soft. Eng. for Adaptive and Self-Managing Systems. pp. 125–134. ACM, New York (2014)
26. Karur, K., Sharma, N., Dharmatti, C., Siegel, J.E.: A survey of path planning algorithms for mobile robots. Vehicles **3**(3), 448–468 (2021)
27. Krarup, B., Cashmore, M., Magazzeni, D., Miller, T.: Model-based contrastive explanations for explainable planning. In: Proc. of the 29th Int'l Conf. on Automated Planning and Scheduling (2019)
28. Lera, F.J.R., Llamas, C.F., Guerrero, Á.M., Olivera, V.M.: Cybersecurity of robotics and autonomous systems: Privacy and safety. Robotics - Legal, Ethical and Socioeconomic Impacts (2017)
29. Li, N., Zhang, M., Li, J., Kang, E., Tei, K.: Preference adaptation: user satisfaction is all you need! In: Proc. of the 18th Symp. on Soft. Eng. for Adaptive and Self-Managing Systems. pp. 133–144 (2023)
30. Liaskos, S., Jalman, R., Aranda, J.: On eliciting contribution measures in goal models. In: Proc. of the 20th IEEE Int'l Requirements Eng. Conf. pp. 221–230 (2012)
31. Lim, W.C.: Managing software reuse: a comprehensive guide to strategically reengineering the organization for reusable components. Prentice-Hall, Inc. (1998)
32. Mahdavi-Hezavehi, S., Durelli, V.H., Weyns, D., Avgeriou, P.: A systematic literature review on methods that handle multiple quality attributes in architecture-based self-adaptive systems. Information and Software Technology **90**, 1–26 (2017)
33. Miller, T.: Explanation in artificial intelligence: Insights from the social sciences. Artificial intelligence **267**, 1–38 (2019)
34. Morandini, M., Penserini, L., Perini, A.: Towards goal-oriented development of self-adaptive systems. In: Proc. of the 2008 Int'l Workshop on Software Eng. for Adaptive and Self-Managing Systems. pp. 9–16 (2008)
35. Paucar, L.H.G., Bencomo, N.: Re-pref: support for reassessment of preferences of non-functional requirements for better decision-making in self-adaptive systems. In: Proc. of the 24th Int'l Requirements Eng. Conf. pp. 411–414. IEEE (2016)
36. Prabhushankar, M., Kwon, G., Temel, D., AlRegib, G.: Contrastive explanations in neural networks. In: Proc. of the IEEE Int'l Conf. on Image Processing (ICIP). pp. 3289–3293. IEEE (2020)
37. Reynolds, O., García-Domínguez, A., Bencomo, N.: Automated provenance graphs for models@ run. time. In: Proc. of the 23rd Int'l Conf. on Model Driven Engineering Languages and Systems: Companion Proceedings. pp. 1–10 (2020)

38. Saaty, R.: The analytic hierarchy process—what it is and how it is used. Mathematical Modelling **9**(3), 161–176 (1987)
39. Samin, H., Bencomo, N., Sawyer, P.: Pri-AwaRE: Tool support for priority-aware decision-making under uncertainty. In: Proc. of the 29th Int'l Requirements Eng. Conf. pp. 450–451. IEEE (2021)
40. Samin, H., Bencomo, N., Sawyer, P.: Decision-making under uncertainty: be aware of your priorities. Software and Systems Modeling pp. 1–30 (2022)
41. Sawyer, P., Bencomo, N., Whittle, J., Letier, E., Finkelstein, A.: Requirements-aware systems: A research agenda for RE for self-adaptive systems. In: Proc. of the 2010 18th IEEE Int'l Requirements Eng. Conf. pp. 95–103. IEEE (2010)
42. Setchi, R., Dehkordi, M.B., Khan, J.S.: Explainable robotics in human-robot interactions. Procedia Computer Science **176**, 3057–3066 (2020)
43. Shaikh, M.T., Goodrich, M.A.: Design and evaluation of adverb palette: A gui for selecting tradeoffs in multi-objective optimization problems. In: Proc. of the ACM/IEEE Int'l Conf. on Human-Robot Interaction. p. 389–397 (2017)
44. Soltani, A.R., Tawfik, H., Goulermas, J.Y., Fernando, T.: Path planning in construction sites: performance evaluation of the Dijkstra, A*, and GA search algorithms. Advanced engineering informatics **16**(4), 291–303 (2002)
45. Sukkerd, R., Simmons, R., Garlan, D.: Tradeoff-focused contrastive explanation for mdp planning. In: Proc. of the 29th IEEE Int'l Conf. on Robot and Human Interactive Communication (RO-MAN). pp. 1041–1048. IEEE (2020)
46. Tipaldi, M., Glielmo, L.: A survey on model-based mission planning and execution for autonomous spacecraft. IEEE Systems Journal **12**(4), 3893–3905 (2017)
47. Wang, H., Yu, Y., Yuan, Q.: Application of Dijkstra algorithm in robot path-planning. In: Proc. of the Int'l Conf. on Mechanic Automation and Control Engineering. pp. 1067–1069. IEEE (2011)
48. Wang, X., Liu, Z., Liu, J.: Mobile robot path planning based on an improved A* algorithm. In: Proc. of the Int'l Conf. on Computer Graphics, Artificial Intelligence, and Data Processing. vol. 12604, pp. 1093–1098. SPIE (2023)
49. Weyns, D., Gerostathopoulos, I., et al.: Preliminary results of a survey on the use of self-adaptation in industry. In: Proc. of the 2022 Int'l Symp. on Soft. Eng. for Adaptive and Self-Managing Systems. pp. 70–76 (2022)
50. Wieringa, R.J.: Design Science Methodology for Information Systems and Software Engineering. Springer (2014)
51. Wohlrab, R., Garlan, D.: A negotiation support system for defining utility functions for multi-stakeholder self-adaptive systems. Requirements Engineering (2021)
52. Wohlrab, R., Meira-Góes, R., Vierhauser, M.: Run-time adaptation of quality attributes for automated planning. In: Proc. of the 17th Symp. on Soft. Eng. for Adaptive and Self-Managing Systems. pp. 98–105 (2022)