Algorithmic transparency in path planning
A visual approach to enhancing human understanding

Zou, Y.; Borst, C.

DOI
10.1016/j.ijhcs.2025.103573

Publication date
2025

Document Version
Final published version

Published in
International Journal of Human-Computer Studies

Int. J. Human–Computer Studies 203 (2025) 103573

Contents lists available at ScienceDirect

International Journal of Human - Computer Studies

journal homepage: www.elsevier.com/locate/ijhcs

Algorithmic transparency in path planning: A visual approach to enhancing human understanding

Yiyuan Zou *, Clark Borst

Control and Simulation, Faculty of Aerospace Engineering, Delft University of Technology, Delft, The Netherlands

ARTICLE INFO

Keywords:
Algorithmic transparency
Path planning
Visualization
User study
Understanding

ABSTRACT

Computer algorithms facilitate increased automation in various human-centered work areas to improve operational safety and efficiency. Algorithmic transparency is considered essential for human operators, policy makers and system developers, as it allows them to understand the capabilities and limitations of an algorithm. In this research, we focus on path-planning algorithms and propose a purely visual approach to achieve their transparency. This approach extracts and portrays information directly from the algorithms, aiming to visually reveal their inner workings. Benchmark tests indicate that extracting information from path-planning algorithms may significantly slow them down. For time-constrained operations, it is recommended to store only the necessary data during the pathfinding process and perform information extraction afterwards. Based on theories from cognitive engineering, six transparency levels were designed to chunk meaningful information pertaining path-planning algorithms. A user study among non-experts ($N = 40$) was then conducted to evaluate the impact of visual algorithmic transparency on human understanding. The results suggest that increased transparency levels allow non-experts to more correctly and confidently understand the details of a path-planning algorithm. However, it is also found that certain transparency levels can lead to confusion, especially when the algorithm behaves in a way contrary to human expectations. This study further reveals that, given the same level of transparency, sampling-based algorithms may be easier to comprehend than graph-based algorithms. This research can serve as a reference for how to achieve transparency in path-planning-related applications and how to hierarchically portray and organize transparency information.

1. Introduction

As technology advances, more and more algorithms are being developed to help people do their jobs more safely and efficiently. However, the complexity of algorithms is generally increasing, making them more difficult for humans to understand. This lack of understanding hampers human trust and acceptance of the advanced technology, thereby limiting its real-world applications (Billings, 1997; Barredo Arrieta et al., 2020; Langer et al., 2021). In the aviation domain, operator acceptance has proven to be one of the largest challenges to successfully introducing new advanced automation (Bekier et al., 2011; Hilburn et al., 2014).

To address this issue, Explainable Artificial Intelligence (XAI) has emerged in recent years, attracting widespread attention from researchers in various fields (Gunning and Aha, 2019; Adadi and Berrada, 2018; Mohseni et al., 2021; Miller, 2019; Endsley, 2023b; SESAR, 2022). XAI is defined as "AI systems that can explain their rationale to a human user, characterize their strengths and weaknesses, and convey an understanding of how they will behave in the future" (Gunning and Aha, 2019). It was originally derived from Explainable Machine Learning (XML) because advanced ML models, especially Artificial Neural Networks (ANN), are typically too complicated to interpret (Doshi-Velez and Kim, 2017; Lipton, 2018). Researchers seek to understand the knowledge embedded within a trained ML model or the decision rules the model acquires via learning. Nowadays, XAI has become a multidisciplinary field (Langer et al., 2021; Mohseni et al., 2021; Miller, 2019), covering AI (Ribeiro et al., 2016; Lundberg et al., 2020), Human-Computer Interaction (HCI) (Lyons, 2013; Amershi et al., 2019) and Cognitive Science (Lombrozo, 2006; Hoffman and Klein, 2017). This expansion is driven by the recognition that understanding AI is beneficial to a wide range of stakeholders, including not only researchers, but also engineers, domain users, and regulators (Barredo Arrieta et al., 2020; Langer et al., 2021). The European Union (EU) General Data Protection Regulation (GDPR) has established the "Right to Explanation" for individuals affected by algorithmic decisions (Goodman and Flaxman, 2017). As AI is a broad topic, this article focuses

* Corresponding author.
E-mail addresses: y.zou@tudelft.nl (Y. Zou), c.borst@tudelft.nl (C. Borst).

https://doi.org/10.1016/j.ijhcs.2025.103573
Received 25 November 2024; Received in revised form 29 April 2025; Accepted 9 June 2025
Available online 30 June 2025

mainly on path planning, exploring a practical approach to enhance its comprehensibility.

Some initial efforts have been made in this direction, such as Explainable AI Planning (XAIP) (Chakraborti et al., 2020; Sreedharan et al., 2019, 2021) and Explainable Motion/Path Planning (Brandão et al., 2021a,c; Kottinger et al., 2021; Brandao et al., 2022; Almagor et al., 2024). *Model Reconciliation* (Sreedharan et al., 2019, 2021) is one of the representative works in this field, which was developed by considering the process of explanations as the reconciliation between human mental models and AI models. There, the model difference, along with inferential limitations of humans, is the root cause of the need for explanations (Chakraborti et al., 2020). However, this approach heavily relies on estimating human mental models (Sreedharan et al., 2018; Tabrez et al., 2020), which may generate meaningless explanations if the mental models are inaccurately estimated. To complement model reconciliation, we are therefore considering "seeing-into" ***transparency*** (Lyons, 2013; Chen et al., 2014; Bhaskara et al., 2020; Springer and Whittaker, 2020; Jamieson et al., 2022). Instead of identifying the model difference, transparency entails revealing all information related to the AI models, making it readily accessible to humans. By reviewing the transparency information and learning from it, humans can also narrow the gap between their mental model and the AI model. The advantage of transparency is that it gives users access to all relevant information. Not only the *output*, but also the *internal process* leading to the output must be disclosed. The downside of transparency, however, is that too much information can become overwhelming and impede understanding.

For path planning, visualization could be an effective way to achieve seeing-into transparency due to its spatio-temporal characteristics that can be intuitively understood. In fact, many pathfinding visualizers have already been developed to visualize various algorithms (Patel, 2023; Sturtevant, 2023; Xu, 2023; Mihailescu, 2023; Misra, 2023; ZJU-FAST-Lab, 2023; Javaid, 2023; Toma et al., 2021). However, most of them either serve as one-off demonstrators (Patel, 2023; Sturtevant, 2023) or are confined to specific domains (Xu, 2023; ZJU-FAST-Lab, 2023). There is no clearly defined, unified method among them. Considering the fact that numerous graph- and sampling-based path-planning algorithms find paths by constructing search trees (Daniel et al., 2010; Harabor et al., 2016; Kavraki et al., 1996; Kuffner and LaValle, 2000; Karaman and Frazzoli, 2011; Gammell et al., 2020), we propose a general approach to extract search trees from the algorithm's internal process for visualization. To validate the effectiveness of this approach, we applied it to over ten representative path-planning algorithms, such as A* (Hart et al., 1968), Theta* (Daniel et al., 2010), Anya (Harabor et al., 2016), Polyanya (Cui et al., 2017), Rapidly-exploring Random Tree (RRT) (Kuffner and LaValle, 2000), RRT* (Karaman and Frazzoli, 2011), Informed RRT* (Gammell et al., 2014), Batch Informed Trees (BIT*) (Gammell et al., 2020), Time-Optimal Any-Angle Safe-Interval Path Planning (TO-AA-SIPP) (Yakovlev and Andreychuk, 2021) and Zeta*-SIPP (Zou and Borst, 2024). Benchmark tests have been conducted to evaluate the influence of search tree extraction on the algorithm runtime.

To avoid overwhelming human users in operational contexts, many studies recommend organizing transparency information hierarchically and providing information on demand (Chen et al., 2014; Bhaskara et al., 2020; Springer and Whittaker, 2020; Zou and Borst, 2023). Rather than just animating algorithms, as existing visualizers do, we designed different levels of path-planning transparency. A new web-based pathfinding visualizer[1] was developed to implement these transparency levels in the path-planning algorithms mentioned above. Based on the visualizer, we further conducted a user study to evaluate the impact of different transparency levels on human understanding. Relevant research regarding the effects of algorithm visualization/animation on understanding has been performed in Computer Science (CS) education (Hundhausen et al., 2002; Kehoe et al., 2001; Naps et al., 2002; Shaffer et al., 2010). However, in CS education, algorithm visualization mainly serves as a supplementary or auxiliary material to the core textbook teaching materials. In our case, we were more interested in the level of understanding achievable solely through visualization. If a low level of transparency could lead to a relatively high level of understanding, perhaps a small amount of information is sufficient to meet user needs, thereby reducing the possibility of information overload in operational scenarios. In our user study, we only invited non-experts to participate because experts with prior knowledge of path planning may confound the results. Users of path-planning-related applications may be the ones who are unfamiliar with path-planning algorithms, such as air traffic controllers and policy makers. The study results could potentially provide a reference for these use cases.

In summary, this research aims to visually reveal the inner workings of path-planning algorithms, organize the disclosed information into distinct levels, and investigate their impact on the understanding of non-expert users. This research could serve as a practical example of operationalizing a multidisciplinary perspective to study XAI and transparency (Langer et al., 2021; Mohseni et al., 2021; Miller, 2019). Here, we first dissected the internal mechanisms of path-planning algorithms and summarized their common steps (AI). Then we designed transparency elements and levels, and integrated them into an interactive path planning visualizer (HCI). Finally, we conducted a user study to investigate the impact of these transparency levels on human understanding (Cognitive Science). While this article specifically addresses path planning, this research method can be generalized to other AI fields for promoting the real-world applications of AI (Endsley, 2023a).

The paper is structured as follows: Section 2 reviews relevant literature on explainable AI and motion/path planning, path planning visualization, design frameworks and guidelines for transparency, and user studies. Section 3 outlines the common steps of graph- and sampling-based path-planning algorithms, laying the groundwork for this research. Section 4 introduces a general approach for information extraction and evaluates its performance based on benchmark testing. Section 5 categorizes algorithmic transparency of path planning into different levels and develops a web-based pathfinding visualizer for its implementation. Section 6 presents a user study for evaluating the impact of algorithmic transparency on human understanding. Section 7 discusses the potential effects and applications of algorithmic transparency, as well as the limitations of this research.

2. Related work

2.1. Explainability and algorithmic transparency

Explainability and algorithmic transparency are highly related concepts, both dedicated to enhancing the comprehensibility of models or algorithms. Explainability refers to the "details and reasons a model gives to make its functioning clear or easy to understand given a certain audience" (Barredo Arrieta et al., 2020), whereas algorithmic transparency indicates the "disclosure of information about algorithms to enable monitoring, checking, criticism, or intervention by interested parties" (Diakopoulos and Koliska, 2017). Explainability is associated with Explainable AI (XAI) and the notion of explanation (Barredo Arrieta et al., 2020). As mentioned in Bitzer et al. (2023), both explainability and algorithmic transparency address "black box" issues. The main difference is that explainability pertains to the "lack of clarity on the reasoning of AI", aiming to providing tailored explanations for why a particular output is generated, while algorithmic transparency concerns the "lack of clarity on the functioning of algorithms", focusing on simply disclosing all relevant information including inputs, outputs and internal processes (Bitzer et al., 2023). Enhancing explainability can actually be seen as a way to boost transparency, as it enables the extraction of more human-understandable information from models,

[1] URL: http://dronectr.tudelft.nl/, ID: pathfinder.

which can then be revealed to humans (Springer and Whittaker, 2020; Rader et al., 2018).

There are three primary strategies for enhancing explainability (Gunning and Aha, 2019). The first strategy involves creating inherently interpretable models, known as “white box” models, exemplified by linear regression and decision trees. The second strategy employs post-hoc methods, utilizing simplified interpretable models to approximate the functionality of the original complex models (“black box”), examples of which include Local Interpretable Model-agnostic Explanations (LIME) (Ribeiro et al., 2016) and SHapley Additive exPlanations (SHAP) (Lundberg and Lee, 2017). The third strategy focuses on developing models with partially explainable features. An example is the decomposed reward Deep Q-Networks (drDQN) that divides rewards into various meaningful components, thereby making the rationale behind actions understandable through the comparison of sub-rewards (Juozapaitis et al., 2019).

Some researchers have raised concerns that the explanations generated by post-hoc methods may not accurately reflect the functioning of the original “black box” models (Rudin, 2019). They argue that such explanations should rather be termed “observations”, because they merely approximate the input–output relationships by observing the black box’s behavior without truly explaining its mechanisms and rationale (Zhu et al., 2018). Therefore, in high-stakes domains, caution is advised when implementing post-hoc methods due to the potential risks arising from discrepancies between the original model and the post-hoc explainable model, and, if possible, “white box” models are preferred (Rudin, 2019).

In this research, we focus on algorithmic transparency instead of explainability, as our main interest is in revealing the inner workings of path-planning algorithms rather than specifically explaining why a certain path is chosen. Since the final path is a direct outcome of the internal process, understanding the process should naturally lead to an understanding of the output. Additionally, the nature of graph- and sampling-based path-planning algorithms allows their internal processes to be relatively easily disclosed, similar to “white box” models in XAI. These prompted us to choose algorithmic transparency as a means to enhance human understanding in path planning.

2.2. Explainable motion and path planning

While XAI predominantly addresses black-box, learning-based approaches, XAIP represents a burgeoning field with a focus on elucidating automated planning and decision-making processes (Chakraborti et al., 2020). Explainable motion/path planning, a specific subfield of XAIP, strives to render the motion/path plans generated by AI more comprehensible to humans. Brandão et al. (2021b) summarized different kinds and purposes of explanations within motion planning and developed optimization-based and sampling-based explainable motion planners. Their attention was not limited to failure explanations (why did you fail?), but extended to contrastive explanations (why choose A over B?). Contrastive map-based explanations were introduced to clarify path plan optimality (Brandão et al., 2021c; Brandão and Setiawan, 2022). These explanations were framed as solutions to inverse-shortest path problems, calculating the minimal modification required to the map to make the user’s desired path optimal. This means if a user inquires why path A is optimal instead of path B, the explainable planner modifies the map to demonstrate scenarios where path B would be optimal. This approach has been extended to the failure-explanation problem in motion planning (Liu and Brandão, 2024). Furthermore, a novel visualization approach based on temporal segmentation for multi-agent motion/path planning was proposed to explain path feasibility (Kottinger et al., 2021; Almagor et al., 2024; Almagor and Lahijanian, 2020). This approach decomposes a plan into segments in which the agents’ paths are disjoint, thereby illustrating that the plan is conflict-free and feasible.

In summary, the existing methods for explainable motion/path planning focus on explaining the optimality and feasibility of the final plan (output), often leveraging contrastive explanations. Referring to this, when disclosing the internal process of a path-planning algorithm for algorithmic transparency, some form of cost-based contrastive explanations can also be integrated to illustrate why the algorithm follows a certain search direction and why an explored path is not the final path. For example, in an A*-based algorithm, the current node has the lowest cost value (f-value) among the nodes in the *open* list. From the cost values of the open nodes, the next current node (next search step) can be easily predicted.

2.3. Path planning visualization

The aforementioned approaches concentrate on elucidating the outputs of motion/path planning rather than the internal mechanisms behind them. Distinct from various ML models, the internal processes of motion/path planning, particularly in 2D environments, lend themselves more readily to visualization. This accessibility has led to the development of numerous pathfinding visualizers, significantly improving the transparency of path planning.

Red Blob Games (Patel, 2023) and Moving AI Lab’s Single Agent Search (SAS) (Sturtevant, 2023) were online tools tailored for educational purposes, featuring interactive demonstrators and accompanying learning resources. These tools offer detailed explanations of classic graph-based path-planning algorithms. PathFinding.js (Xu, 2023) and Mihailescu’s Pathfinding Visualizer (Mihailescu, 2023) were primarily developed for graph-based path-planning, offering many adjustable options to modify the inputs and settings. Upon execution, an animation that illustrates the search process and highlights the optimal path will be displayed. Pathfinding Visualizer ThreeJS (Misra, 2023) attempts to portray several classic graph-based path-planning algorithms in 3D environments. While the algorithms are designed for 2D, the perspective can be smoothly transformed into 3D. This feature has the potential to be integrated into map navigation. FAST Lab’s Sampling-based Visualizer (ZJU-FAST-Lab, 2023) and Local Planner Visualization Project (LPVP) (Javaid, 2023) concentrate on sampling-based path-planning algorithms rather than graph-based ones, presenting the exploration processes of search trees through animation. LPVP can also portray potential-field-based path-planning algorithms by presenting their vector fields. PathBench (Toma et al., 2021) is an integrated platform designed for portraying graph-based, sampling-based and learning-based path-planning algorithms, supporting both 2D and 3D environments. It also includes an evaluation module for benchmarking different path-planning algorithms. Posthoc (Zheng et al., 2024) is a debugging and visualization framework for search algorithms. It takes as input a structured log describing basic search operations and provides a visual interface used for playback, analysis and visualization.

Considering the goal of this research is to achieve algorithmic transparency in path planning, we primarily draw inspiration from pathfinding visualizers rather than explainable path planning methods. Instead of providing specific explanations for the output, our approach aims to disclose all relevant information, allowing users to form their own judgments. By presenting how the algorithm works, users could understand how it navigates within the map, what constraints it takes into account, and why it generates a specific path. Due to the benefits of visualization in presenting time-related procedural knowledge (Hundhausen et al., 2002; Aysolmaz and Reijers, 2021; Tversky et al., 2002) and various successful practices in pathfinding visualizers, we also decided to adopt a visual approach rather than a textual or verbal approach for designing transparency. However, the existing visualizers do not systematically illustrate how to dissect path-planning algorithms (e.g., common steps or a unified framework) for visualization and/or offer a way to hierarchically organize information for helping non-expert users. This research aims to fill this gap.

2.4. Design frameworks and guidelines

From a user-centered perspective, explanatory and transparency information should be presented in accordance with user demands, limitations, preferences and needs. Several user-centered frameworks have been proposed to design XAI systems (Mohseni et al., 2021; Zhu et al., 2018; Eiband et al., 2018; Wang et al., 2019). Wang et al. (2019) proposed a theory-driven user-centric XAI framework, inspired by the theoretical underpinnings of human reasoning. This framework aims to support human reasoning processes and reduce cognitive biases by providing tailored explanations. Instead of abstract design guidelines, Eiband et al. (2018) presented a practical approach with their stage-based participatory design process for achieving transparency by explanation interface design. This process was divided into two pivotal phases: identifying the content of explanations and determining the best approach to deliver them, thus offering concrete guidance for designers. Building upon these foundations, Mohseni et al. (2021) developed a new nested XAI framework that integrates both design and evaluation phases. This comprehensive framework encompasses three layers: the overall XAI system, the explanation interface, and the core explainable models and algorithms. The design process of this framework initiates with defining the goals of the XAI system in the outer layer, progresses by tailoring the interface to meet user needs in the middle layer, and finally focuses on the underlying algorithms in the innermost layer. In each layer, the stages of design and evaluation construct an iterative cycle.

Another line of research related to user-centered AI systems is Automation and Agent Transparency (Bhaskara et al., 2020), such as the belief–desire-intention (BDI) framework (Rao and George, 1995), Lee and See's 3Ps (Purpose, Process and Performance) theory in human trust in automation (Lee and See, 2004), Lyons' human–robot transparency model (Lyons, 2013) and Chen's situation awareness-based agent transparency (SAT) model (Chen et al., 2014). To avoid overwhelming human users, transparency is generally divided into different levels, enabling a progressive and incremental disclosure of information (Springer and Whittaker, 2020). For example, the SAT model contains three levels (Chen et al., 2014, 2018) : Level 1 - Basic Information (agent's current status/actions/plans), Level 2 - Rationale (agent's reasoning process) and Level 3 - Outcomes (agent's projections/predictions; uncertainty), each corresponding to and supporting the three levels of operator situation awareness (SA) respectively (Endsley, 1995). More relevant reviews can be found in Endsley (2023b), Bhaskara et al. (2020) and van de Merwe et al. (2024).

Although many transparency frameworks were developed for multi-agent systems, they can also serve as a reference for the algorithmic transparency design. These frameworks often divide transparency into different components and organize them into a hierarchical structure. As deeper information is revealed, a higher level of understanding could be achieved. As such, this research also explores how to divide path-planning transparency into different elements and how to design transparency levels to progressively disclose information about path-planning algorithms.

2.5. User studies

To evaluate the impact of XAI and transparency on user perceptions, understanding, acceptance and trust, several user studies and human-in-the-loop experiments have been conducted (Mohseni et al., 2021; Bhaskara et al., 2020; van de Merwe et al., 2024; Vilone and Longo, 2021). However, the reported results are mixed (Springer and Whittaker, 2020; Chen et al., 2023). Some studies suggest clear positive effects of transparency (Rader et al., 2018; Dzindolet et al., 2003; Lim et al., 2009; Nothdurft et al., 2014; Stowers et al., 2020; Cheng et al., 2019) while others do not (Bunt et al., 2012; van der Waa et al., 2021). These findings indicate that additional factors influence the effectiveness of transparency, such as explanation modalities (Schmude et al., 2023), meaningfulness (Nourani et al., 2019), information amount (van der Waa et al., 2021; Kizilcec, 2016), user expectations (Springer and Whittaker, 2020; Kocielnik et al., 2019; Riveiro and Thill, 2021) and domain knowledge (Dikmen and Burns, 2022). Designing for transparency should take these factors into account as well. Interestingly, investigations in CS education have also identified similar mixed results regarding the effectiveness of algorithm visualization (Gurka and Citrin, 1996). Through a meta-study (Hundhausen et al., 2002) and a series of experiments (Grissom et al., 2003), researchers found that *engagement* (e.g., interacting with the algorithm) plays a vital role in learning and understanding (Naps et al., 2002; Grissom et al., 2003).

To evaluate the proposed approach for transparent path planning, a user study is also required. Considering the central importance of understanding in transparency (Langer et al., 2021), the user study in this article was specifically designed to explore the impact of transparency on human understanding instead of trust and task performance. Additionally, previous user studies have rarely assessed the understanding of planning processes, focusing mainly on classification and prediction problems (Schmude et al., 2023), or planning outcomes (final plans) (Sreedharan et al., 2021). This study could fill this gap and serve as an example of how to conduct such an evaluation. To avoid potential bias from experts' prior knowledge, only non-experts were invited to participate. An interactive interface was implemented to enhance participant engagement (Cheng et al., 2019), allowing them to freely explore the algorithm's behavior and maximize the use of available information at different transparency levels.

3. Preliminaries

In this section, we will review various graph- and sampling-based path-planning algorithms, identifying and summarizing the common steps among them. It will provide a foundation for designing algorithmic transparency, including information extraction and visualization.

3.1. Graph-based path planning

Since the continuous space contains infinite points, graphs are usually defined to limit the complexity of the search space S for path planning. Graph-based path planning firstly generates a graph $G = (V, E, C_E)$ where V is a set of vertices, $E \subseteq V \times V$ is a set of edges (pairs of vertices) and $C_E : E \to \mathbb{R}^+$ is a cost associated with each edge. Then a search algorithm can be applied to search for the optimal solution within the limited search space. One benefit of such a graph is that the connections between vertices are predefined before searching, and thus it is easier to expand nodes when creating search trees. Without the predefined edges, the algorithm has to perform line-of-sight checks to expand nodes and build branches, which is generally more time-consuming for the search. There are many different approaches to generate graphs, such as regular grids, navigation meshes and visibility graphs (Lawande et al., 2022). Regular grids are perhaps the most common approach for discretizing continuous spaces. The graph vertices on regular grids can be either the grid vertices or the grid centers. Different from regular grids, navigation meshes are collections of convex polygons that can more accurately represent the environment. Regular grids can be viewed as a special case. Visibility graphs are graphs whose edges represent visibility connections between corresponding vertices. They are usually constructed from the vertices of obstacles.

A* is one of the most classic graph-based path-planning algorithms (Hart et al., 1968). It has been applied to various fields and still forms a basis for many advanced algorithms (Harabor et al., 2016; Hechenberger et al., 2020). However, traditional A*-based algorithms are mainly implemented on regular grids to find grid-by-grid paths. The connections for each grid are constructed only between it and its eight adjacent grids. This kind of search is limited to 45-degree increments and thus the paths are not the truly shortest (Daniel et al., 2010).

Algorithm 1 A*-based path planning

Input: Search space S, start p_s, target p_t
Output: Optimal path ρ
1: Graph $G \leftarrow$ `discrete`(S)
2: Start node $n_s \leftarrow$ `createNode`(p_s, G)
3: Tree nodes $N \leftarrow \{n_s\}$, $open \leftarrow \{n_s\}$
4: **while** $open \neq \emptyset$ **do**
5: $\quad n_c \leftarrow \arg\min_{n \in open} f(n)$
6: $\quad open \leftarrow open \setminus \{n_c\}$
7: $\quad$ **if** `isReach`(n_c, p_t) **then**
8: $\quad\quad$ **return** `pathTo`(n_c)
9: $\quad$ **end if**
10: $\quad N_c \leftarrow$ `successors/predecessors`(n_c, G)
11: $\quad open \leftarrow open \cup (N_c \setminus N)$, $N \leftarrow N \cup N_c$
12: $\quad$ Rewire the search tree and update the node costs
13: **end while**
14: **return** *null*

Therefore, any-angle path planning is introduced to address this issue, such as Theta* (Daniel et al., 2010), Block A* (Yap et al., 2011) and Anya (Harabor et al., 2016), which removes the 45-degree limitation to generate straighter and shorter paths. TO-AA-SIPP (Yakovlev and Andreychuk, 2021) and Zeta*-SIPP (Zou and Borst, 2024) are optimal any-angle path-planning algorithms combined with Safe Interval Path Planning (SIPP) (Phillips and Likhachev, 2011) for dynamic environments. In addition to any-angle path planning on grids, many mesh-based and visibility-graph-based algorithms are also designed to find true shortest paths (Shen et al., 2022). For example, Polyanya (Cui et al., 2017) is an extension of Anya, which is applied on navigation meshes instead of regular grids. RayScan (Hechenberger et al., 2020) is an online path-planning algorithm which partially builds a visibility graph by using ray shooting to discover obstacles and then scanning along their edges to find ways around them.

In general, despite the variety of graph-based path-planning algorithms, most are still based on the A* search. The algorithms mentioned above are all A*-based. Therefore, we summarize the main procedure of A*-based path planning, as shown in Algorithm 1. Here, *open* is a queue, sorted by the f-value in Line 5. The tree nodes N refer to the nodes explored by the algorithm, which can be used to extract the search tree at each step. More details on the usage of N are provided in the next section.

3.2. Sampling-based path planning

Another way to search continuous spaces is by sampling. Rather than predefining a graph, sampling-based algorithms *randomly* pick points in the search space for discretization. The sampling points can be regarded as the vertices of an *implicit* Random Geometric Graph (RGG) (Penrose, 2003) whose edges are more flexible than fixed graphs. Sampling-based path planning can be viewed as constructing an *implicit* RGG and an *explicit* search tree in the search space S (Gammell et al., 2015). There are two main approaches to build edges in RGGs: connecting each vertex to a specific number of its nearest neighbors (Xue and Kumar, 2004) (k-nearest RGGs), or to all its neighbors within a certain distance (Gilbert, 1961) (r-distance RGGs).

According to the usage of RGGs, there are two different types of sampling-based path-planning algorithms: probabilistic roadmaps (PRM) (Kavraki et al., 1996) and RRT (Kuffner and LaValle, 2000). In PRM-based algorithms, a *complete* PRM/RGG with edges is firstly built by sampling after which a search algorithm is applied to find

Algorithm 2 RRT-based path planning

Input: Search space S, start p_s, target p_t, termination condition T
Output: Optimal/Near-optimal path ρ
1: Tree nodes $N \leftarrow \emptyset$ and paths $P \leftarrow \emptyset$
2: Initial condition $t \leftarrow$ `init()`
3: **while** $t \neq T$ **do**
4: $\quad$ Random point $p_i \leftarrow$ `sample`(S)
5: $\quad$ Nearest node $n_{\text{nearest}} \leftarrow$ `nearest`(N, p_i)
6: $\quad$ Search node $n_i \leftarrow$ `steer`$(n_{\text{nearest}}, p_i)$
7: $\quad$ **if** `collisionFree`$(n_{\text{nearest}}, n_i)$ **then**
8: $\quad\quad N \leftarrow N \cup \{n_i\}$
9: $\quad\quad$ Rewire the search tree and update the node costs
10: $\quad\quad$ **if** `isReach`(n_i, p_t) **then**
11: $\quad\quad\quad P \leftarrow P \cup$ `pathTo`(n_i)
12: $\quad\quad\quad S \leftarrow$ `updateSearchSpace`(P, S)
13: $\quad\quad$ **end if**
14: $\quad$ **end if**
15: $\quad t \leftarrow$ `update`(t)
16: **end while**
17: **return** `bestPath`(P)

the optimal path. The main difference between PRM-based and graph-based path planning is the method of generating graphs. The former is based on sampling within a local region while the latter is a decomposition of the entire search space. BIT* (Gammell et al., 2020, 2015) can be viewed as a variant of PRM-based algorithms. For each *batch* in BIT*, an implicit RGG is built (or revised) and then a heuristic search can be applied to incrementally expand the search tree. Different from PRM-based algorithms, RRT-based algorithms do not directly connect sampling points to build edges in RGGs, but instead utilize the sampling points to provide expansion *directions* for search trees. In each iteration, after sampling a random point in the search space, the RRT-based algorithm finds the nearest node in the search tree to this random point, and then extends a new branch from the nearest node to it. A parameter, incremental distance (Kuffner and LaValle, 2000), limits the length of this new branch. If the incremental distance is large enough, the sampling points may directly become the nodes of the search tree after sampling and light-of-sight checks. In RRT* (Karaman and Frazzoli, 2011), by checking if the newly added node is a better parent for its neighbors with a certain distance, the search tree can be rewired to guarantee finding asymptotically optimal paths.

In summary, PRM-based path planning also applies search algorithms after building RGGs and the A* search used can also be described by Algorithm 1. Therefore, we only summarize the procedure of RRT-based path planning, as shown in Algorithm 2. The basic ideas of RRT* (line 9) and Informed RRT* (line 12) are also integrated.

3.3. Common steps in path planning

Based on the analysis above, even though graph-based and sampling-based path-planning algorithms appear to be totally different, both of them are essentially based on search algorithms to build search trees. Their main difference is how they create a graph to discretize the search space. Graph-based path planning predefines a graph based on the decomposition of the search space, which is usually fixed but has an efficiently ordered nature. Sampling-based path planning creates an *implicit* RGG by sampling, which has a good scalability and can be continuously improved by adding more samples. After discretizing the search space, a search algorithm, like Dijkstra, A* and RRT, can be applied to generate a search tree and find a feasible or optimal path. In

RRT-based algorithms, the vertices of the RGG are only used to provide expansion *directions* for the search tree.

Therefore, the common steps in graph- and sampling-based path planning can be summarized as follows:

(1) *Define a search space to limit the search process*. The search space S is usually the entire map if there is no other physical constraint related to agent performance, such as vehicle endurance and maneuver limitations. For instance, in grid-based path planning, it is common to include barriers or walls around the map to constrain the search and prevent it from extending beyond the specified boundaries. In Informed RRT* (Gammell et al., 2014), an elliptical region is defined to narrow the search space according to the best path it has found.

(2) *Generate a graph to discretize the search space*. Since the search space is continuous and has an infinite number of points, it is difficult to search directly in the search space. Both graph- and sampling-based path-planning algorithms generate a graph to discretize the search space. Then a search can be performed to find a path.

(3) *Apply a search algorithm to find a path based on the graph*. The real output of a search is a search tree and the final path can be obtained by extraction. There are four main elements in a search algorithm: *search nodes*, *successors or predecessors*, *cost function* and *search procedure*. The successors or predecessors indicate the reachable connections of nodes and thus can be used to generate the branches of search trees. The cost function is designed to evaluate the cumulative costs of nodes, and the search procedure describes the main search steps. Please note that the search nodes are not limited to points and can also take other forms, like lines or triangles.

The diversity of path-planning algorithms are reflected by the different approaches adopted in the common elements. For example, different methods to graph generation in graph-based path planning lead to either regular-grid-based, mesh-based or visibility-graph-based algorithms. The algorithms can generate search trees by using forward expansion to find successors or backward expansion to generate predecessors (Yakovlev and Andreychuk, 2021). How to define an efficient cost function to accurately reflect the goal of path planning is also an important element in real-world applications.

4. Information extraction

As the essence of visualization is information presentation, this section introduces a general approach for extracting information in path planning based on the common steps summarized above. The information extraction is performed during the search process. Depending on when the data is transmitted from the algorithm to the interface within the system, the visualization can be either real-time (during the search) or post-hoc (after the search). Our proposed approach supports both since the data transmission is independent of the data extraction. In this research, we chose post-hoc visualization, which is displayed only when users request more transparency.

4.1. Search tree extraction

As discussed in Section 3, search trees are at the heart of graph-based and sampling-based path-planning algorithms. The final path represents only one of many possible solutions explored by search trees. However, to save memory, the search trees are typically not stored directly during the execution of path-planning algorithms. Only the "latest" or "current" search tree is retained in a specific structured format. Path planning can be regarded as a process of continuously updating this "current" tree until it contains the optimal path. In practice, the tree data is usually embedded within the variables of tree

Algorithm 3 Extract i-th search tree directly

function `directExtract`(N_i)
  Search tree $T_i \leftarrow \varnothing$
  for each $n \in N_i$ **do**
    if $parent(n) \neq null$ **then**
      $T_i \leftarrow T_i \cup \{\ parent(n), n\ \}$
    end if
  end for
  return T_i
end function

Algorithm 4 Extract i-th search tree from branch sets

function `forwardExtract`(Γ_i)
  Search tree $T_i \leftarrow \varnothing$
  for $j = 1, 2, \cdots i$ **do**
    Branches $L_j \leftarrow \Gamma_i(j)$
    for each $\{\ parent(n), n\ \} \in L_j$ **do**
      $T_i \leftarrow T_i \setminus \{\ oldparent(n), n\ \}$
      $T_i \leftarrow T_i \cup \{\ parent(n), n\ \}$
    end for
  end for
  return T_i
end function

function `backwardExtract`(Γ_i)
  Search tree $T_i \leftarrow \varnothing$
  for $j = i, i-1, \cdots 1$ **do**
    Branches $L_j \leftarrow \Gamma_i(j)$
    for each $\{\ parent(n), n\ \} \in L_j$ **do**
      if $connected(n) = false$ **then**
        $T_i \leftarrow T_i \cup \{\ parent(n), n\ \}$
        $connected(n) \leftarrow true$
      end if
    end for
  end for
  return T_i
end function

nodes N (see Algorithms 1 and 2), requiring further extraction from them. To effectively extract search trees, we introduce two methods that specify the necessary data to be stored, along with procedures for extraction, as shown in Algorithms 3 and 4.

Algorithm 3 is perhaps the most straightforward method to extract search trees. The collection of tree nodes N is stored at each search step. By constructing branches from the tree nodes, the search tree at a certain step can be extracted. The search process can then be portrayed by displaying the extracted search trees in order.

However, storing all search trees using Algorithm 3 is not efficient. Much redundant data may be recorded because most nodes and branches are fixed after construction. For example, in A*-based algorithms, when a node n is inserted into the $closed$ list, the branch connecting $parent(n)$ and n is fixed and only needs to be extracted once, as shown in Fig. 1. The changes to the search tree are primarily caused by rewiring among the explored nodes or expansion to new nodes. It is only necessary to record the new changes or new branches at each step. Let L_i denote the new branches at step i, and thus the cumulative

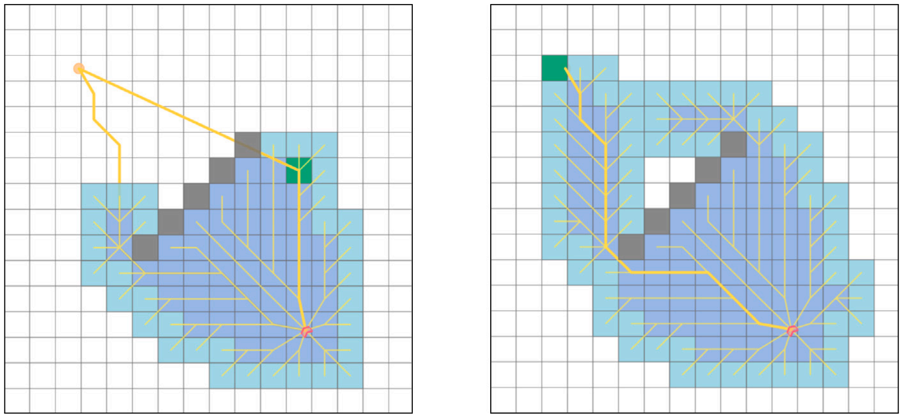


Fig. 1. Part of the search tree is fixed during the search process. The light blue grids indicate the open nodes, while the dark blue (cornflower blue) grids denote the closed nodes. The yellow lines are the branches of search trees and the light orange thick line is the current optimal path found by the algorithm.

collection of new branches at step i is $\Gamma_i \leftarrow \Gamma_{i-1} \cup L_i$. Based on Γ_i, the extraction process for search trees is slightly different. Algorithm 4 introduces forward and backward extraction methods based on Γ_i. Lines 17 and 19 ensure that the new branches will always be connected first and then the old useless branches will be discarded. The forward extraction method is particularly suitable for incremental data extraction as it extracts search trees according to the previous tree (T_{i-1}) and new branches (L_i). The backward method is more efficient for extracting only one search tree since it requires no pruning. Using Algorithm 4, only necessary data Γ_i needs to be recorded during the search process. The benefit of Algorithm 4 is that it separates data storing from data extraction, meaning that it can obtain solutions first and then extract data later for visualization.

In summary, Algorithm 3 has to extract and store all search trees during the search process, which could significantly reduce the real-time performance of path-planning algorithms. In contrast, Algorithm 4 only needs to record new branches at each step, allowing the extraction of search trees to be performed after the search.

In practice, some other useful data can also be recorded for visualization:

(1) *Current node*. The current node is either the most promising node so far, or a new node added to the search tree.
(2) *The path through the current node*. The current path is usually constructed by the explored path from the start to the current node and the expected path from the current node to the target.
(3) In A*-based algorithms, the *status of a node*, whether open or closed, can be recorded and subsequently represented with different colors in the visualization (Xu, 2023).
(4) In RRT-based algorithms, one can record (i) which node is the nearest node to the current node, (ii) the current sampling space, (iii) the current optimal path, (iv) and the rewiring radius.

4.2. Performance analysis

To validate the applicability of the information extraction approach, we applied it to over ten representative path-planning algorithms, such as A*, Theta*, Anya, Polyanya, RRT, RRT*, Informed RRT*, BIT*, TO-AA-SIPP and Zeta*-SIPP. Considering that the SIPP-based algorithms are tailored for dynamic environments, our analysis concentrates on the first eight algorithms to evaluate the impact of information extraction on their performance. To facilitate comparison, all search trees during the search processes were stored and extracted via both Algorithms 3 and 4. The additional runtime introduced by information extraction could be identified and analyzed.

The experiments were conducted on four benchmark sets, including around 0.23 million instances (Sturtevant, 2012). Two of them are from computer games; Dragon Age: Origins and Baldur's Gate II (Original maps). Another one consists of thirty 256 × 256 city/street maps from ten different cities and the final one comprises ten 512 × 512 grid maps, each featuring 10% random obstacles. In the experiments, the maps "den510d" and "orz100d" had to be excluded from the Dragon Age set because the RRT-based algorithms were unable to successfully solve all the problems of the maps within an acceptable time (given fixed parameter settings). For example, the first 1200 instances of the map "den510d" took RRT 18 days to calculate for both path planning and information extraction. This is because the feasible paths in many instances of "den510d" and "orz100d" need to pass through some extremely narrow passages and exits. The RRT-based algorithms are unable to handle this situation efficiently without optimizing their parameters (Tahirovic and Ferizbegovic, 2018; Wu et al., 2021). As the objective is not to evaluate the performance of different path-planning algorithms, the exclusion of these maps is unlikely to have a significant effect on the conclusion drawn from the benchmark tests.

The algorithms were implemented in JavaScript and performed on Node.js v18.14.2 on a laptop with 2.30 GHz Intel Core i7-11800H and 16 GB RAM. For simplicity, rectangular meshes were generated for Polyanya by greedily merging unobstructed grids (Cui et al., 2017). The iteration of the sampling-based algorithms was set to infinity, and they were terminated only when finding a feasible path. In this case, Informed RRT* degenerated into RRT* because the *informed* procedure in Informed RRT* would be activated only upon the discovery of an initial solution. The implemented BIT* is its basic version (Gammell et al., 2015). The performance of sampling-based algorithms heavily depends on their parameter settings. Optimization of these parameters remains an area of ongoing research (Cano et al., 2018). Drawing on the parameter settings from previous studies, the rewiring radius of (Informed) RRT* and BIT* was computed by $2\eta(1+1/d)^{1/d}(\lambda(X_{free})/\zeta_d)^{1/d}(\log(q)/q)^{1/d}$ (Karaman and Frazzoli, 2011; Gammell et al., 2015) where $\eta \geq 1$ is a tuning parameter, d is the dimension of the space, $\lambda(X_{free})$ is the Lebesgue measure of the obstacle-free space, ζ_d is the volume of the unit ball in the d-dimensional space and q is the number of tree nodes. Based on Gammell et al. (2014, 2015), η was set to 1.1. The incremental distance of the RRT-based algorithms (including RRT) was set equal to the rewiring radius (Gammell et al., 2014) and the target range was 5% of $\sqrt{S}$ where S is the area of the entire search space (Gammell et al., 2015). The batch size of BIT* was set to 100 (Gammell et al., 2015).

Tables 1–3 present the results concerning the additional runtime introduced by the information extraction methods. To facilitate analysis and comparison, Fig. 2 summarizes the results across the different maps. These results indicate that, regardless of Algorithms 3 and 4, extracting all search trees from the entire search process could significantly affect the original path-planning runtime, especially for A*

Table 1
Mean ratio of additional runtime to original runtime due to `directExtract` of Algorithm 3.

Benchmark	A*	Theta*	Anya	Polyanya	RRT	(Informed) RRT*	BIT*
Dragon Age	68.9528	26.9459	4.4505	1.8574	0.3315	0.0798	0.0075
Baldur's Gate II	15.0701	6.1994	0.7060	0.3027	0.3476	0.0996	0.0091
City/street maps	23.9722	8.6318	0.6552	0.3663	0.3742	0.0688	0.0069
Random 10%	52.3331	15.5591	10.5738	4.4334	0.4369	0.1326	0.0033

Table 2
Mean ratio of additional runtime to original runtime due to recording Γ for Algorithm 4.

Benchmark	A*	Theta*	Anya	Polyanya	RRT	(Informed) RRT*	BIT*
Dragon Age	0.2002	0.0864	0.0234	0.0089	0.0172	0.0090	0.0016
Baldur's Gate II	0.1725	0.0836	0.0248	0.0071	0.0373	0.0193	0.0030
City/street maps	0.2480	0.1102	0.0202	0.0067	0.0302	0.0114	0.0023
Random 10%	0.1775	0.0929	0.0340	0.0077	0.0022	0.0015	0.0003

Table 3
Mean ratio of additional runtime to original runtime due to `forwardExtract` of Algorithm 4.

Benchmark	A*	Theta*	Anya	Polyanya	RRT	(Informed) RRT*	BIT*
Dragon Age	30.2517	12.6191	2.1679	0.8175	0.1667	0.0640	0.0051
Baldur's Gate II	6.6391	2.8711	0.3197	0.1333	0.1972	0.0790	0.0083
City/street maps	9.4963	4.0004	0.2949	0.1657	0.2168	0.0550	0.0070
Random 10%	21.5124	6.4739	5.6418	2.2755	0.2276	0.0819	0.0015

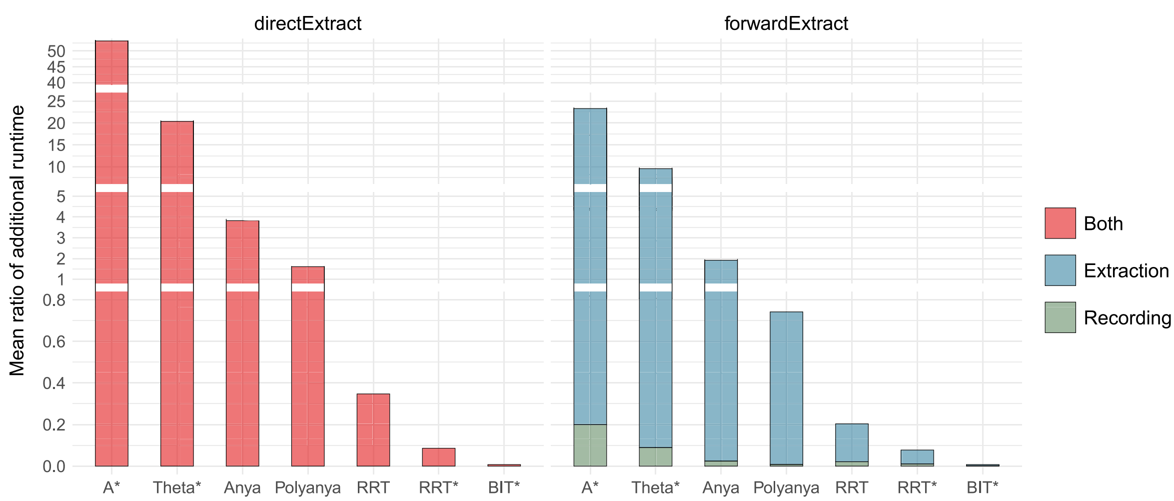


Fig. 2. Comparison of Algorithms 3 and 4 in data recording and extraction.

and Theta*. This is because A* and Theta* both expand search nodes grid by grid and thus construct many branches during their search processes. However, Theta* straightens its search tree by checking if the current node's adjacent nodes can be reached from the current node's parent node and generally constructs fewer branches compared to A* in the same scenario. Therefore, the extraction of search trees has a relatively smaller impact on Theta* compared to A*. In contrast to A* and Theta*, Anya and Polyanya expand their search nodes (triangles) row by row or polygon by polygon, largely reducing the need to build connections (branches) between nodes. Unlike graph-based path-planning algorithms, sampling-based algorithms are less affected by information extraction. This is probably because they are not constrained by predefined graphs and can find feasible paths through fewer branches. Please note that the results in the tables are also affected by the original runtime. The slower the algorithm, the smaller the impact of extracting the same search tree. For example, RRT* is much slower than RRT due to its rewiring calculations, which results in a smaller impact from information extraction. The mean ratios of additional runtime for BIT* are very small because BIT* consumes considerable time constructing RGGs, which are not considered in the search tree extraction. Further research could combine graph preprocessing with search tree extraction to provide a more comprehensive presentation of path planning. As this article focuses on the search tree, which is central to path planning, the construction process of graphs is beyond the scope.

The results of Algorithms 3 and 4 are similar. However, recording only the necessary data Γ will not significantly slow down the original path-planning algorithms. Through Algorithm 4, the search trees can be extracted as needed after the path has been found. The results of this benchmark test also imply that extracting all search trees to present animations may be impractical in real-time, large-scale environments. When computing resources are limited, achieving full transparency could significantly slow down algorithms. As a result, users will receive delayed feedback and information from algorithms, leading to slower responses. In operational contexts with time constraints, this could be problematic and it may be necessary to consider lower levels of transparency (extracting less information). Designing for transparency requires considering both human (cognitive load) and machine (computing resources) limitations.

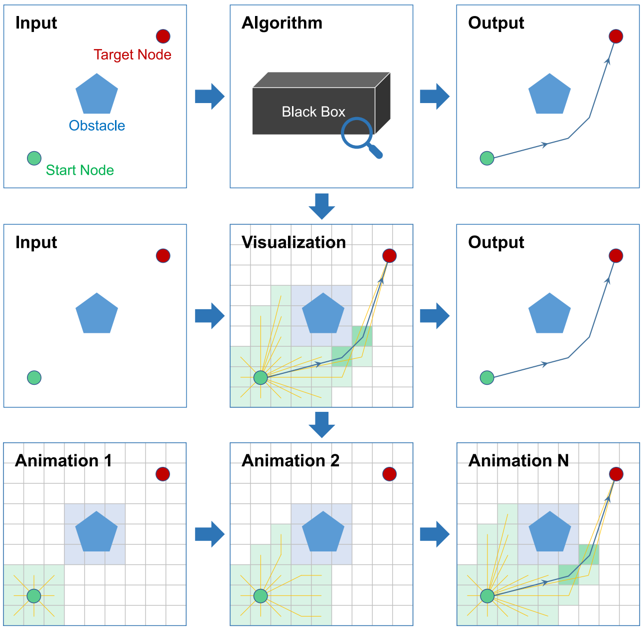


Fig. 3. The overview of path-planning visualization.

5. Information presentation

Drawing on the information extracted, this section proposes an approach for information presentation to achieve algorithmic transparency in path planning. Essentially, the portrayal of path-planning algorithms can be depicted by search trees comprising both explored nodes and constructed branches, as illustrated in Fig. 3. While this type of presentation is prevalent for sampling-based path-planning algorithms (ZJU-FAST-Lab, 2023; Javaid, 2023), surprisingly, it is uncommon for grid-based path planning. Generally, the visualization of grid-based path planning focuses on presenting explored nodes (i.e., grids), neglecting to showcase the expanding, rewiring and pruning processes of search trees (Xu, 2023; Mihailescu, 2023). In this article, we highlight the central importance of search trees in pathfinding visualization and algorithmic transparency.

5.1. Transparency levels

For the purposes of learning and auditing, it may be helpful to have browsable pathfinding animations that offer comprehensive insights into the inner workings of the algorithm. However, in operational contexts, presenting excessive information at the same time may overwhelm human operators (Bhaskara et al., 2020). As suggested by Springer and Whittaker (Springer and Whittaker, 2020), progressive disclosure may be needed for algorithmic transparency, meaning that transparency information needs to be organized hierarchically from ‘simple’ to ‘complex’. To date, however, no clear design guidelines exist for what type of hierarchy would be needed and what ‘simple’ and ‘complex’ entails. Therefore, to clarify transparency in path planning, we firstly designed transparency elements according to a transparency taxonomy proposed in Zou and Borst (2023), as outlined in Table 4. Each element represents a distinct piece of information related to solutions. Since the algorithms reviewed in this article all aim to identify the shortest paths without involving a specific operational context, the categories “Purpose/Intent” and “Expected Outcomes” have been omitted. Subsequently, by reorganizing the transparency elements, six levels of transparency were designed, as shown in Fig. 4. Inspired by Rasmussen’s Abstraction Hierarchy (AH) used in Cognitive Systems Engineering and Ecological Interface Design (EID) (Vicente and Rasmussen, 1992; Rasmussen and Vicente, 1989; Rasmussen, 1985), the transparency levels were organized from functional purpose (i.e., solution) to inner physical structure (i.e., search process), following a typical human top-down, problem-solving strategy. As the transparency level increases, more details about the algorithm’s inner structure are revealed in a cumulative way. Each subsequent, deeper level introduces additional information on top of the preceding levels without excluding any previously disclosed information.

A general example of the transparency levels for two different types of path-planning algorithms is presented in Fig. 5. At Level 0, only the solution is presented. The dashed line indicates the direct path without obstacles. The current optimal path will cause a 5% additional cost compared to the dashed direct path (the numbers are illustrative only). At Level 1, the search space is delineated by an elliptical region, which could be more complex in real-world scenarios (more constraints). Here, the elliptical region can represent the endurance or required time of arrival for moving vehicles (van Paassen et al., 2018). At Level 2, regular grids and sampling points that discretize the search space are showcased. In PRM-based algorithms, the connections between

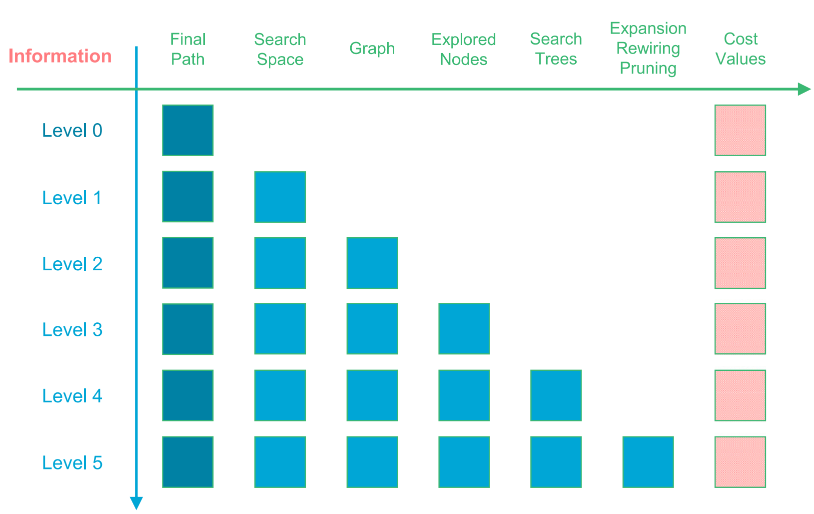


Fig. 4. Proposed transparency levels for path planning.

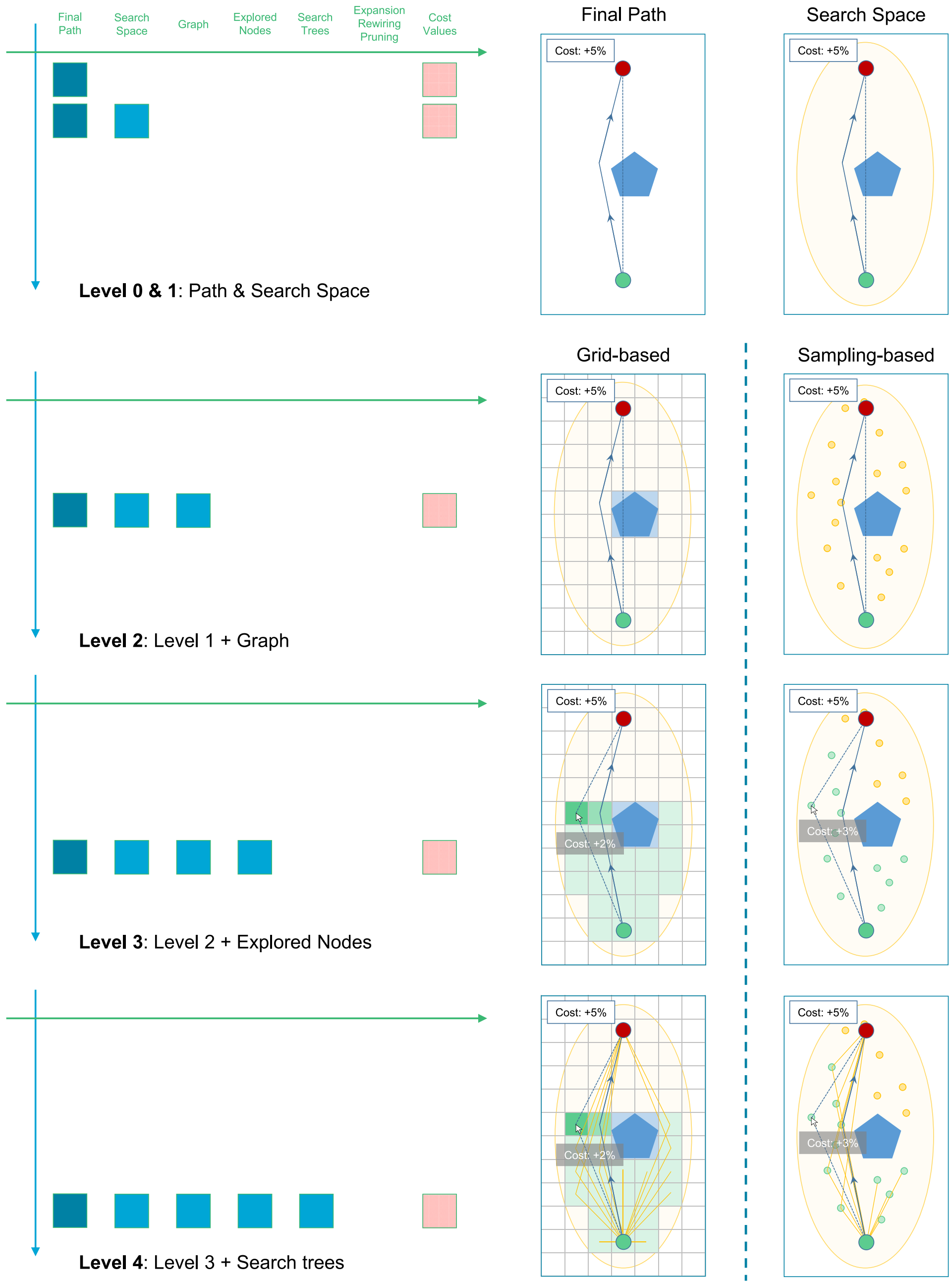


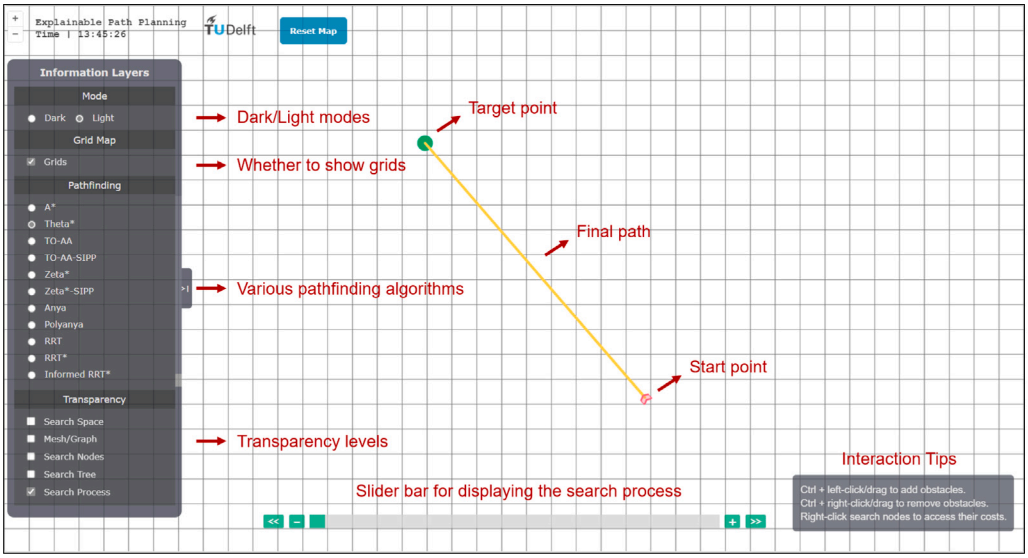
Fig. 5. Implementation of the transparency levels.

Table 4
Proposed transparency elements in path planning.

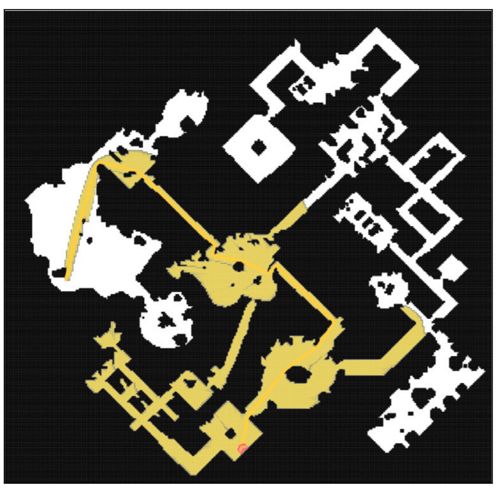
Transparency category	Transparency element
Solution	Final path found by the algorithm
Domain constraints	Search space defined by domain constraints
Explored solutions	Graph: discretized search space Explored nodes: explored search space Search trees: explored potential paths
Cost function/values	Cost values of explored nodes/paths
Computational process	Expansion: explore more space Rewiring: reconnect nodes for shorter paths Pruning: reduce useless nodes and branches

sampling points can also be highlighted. At Level 3, the explored space is illustrated through the display of explored nodes. Users are able to probe the explored space, for example by clicking on a node, to obtain a cost-based contrastive explanation (Miller, 2019) for why the optimal path is *A* rather than *B*. Please note that this click will not trigger a new search but will retrieve the explored path through the clicked node. At Level 4, the search tree is shown to provide more detailed information about the explored solution space. From the figure, one can see many other feasible solutions besides the optimal one. It also provides users with the possibility to change the optimal path to a non-optimal, but maybe a more preferable path, allowing users to intervene in the algorithm based on the existing information (Mohseni et al., 2021; Brandão et al., 2021a).

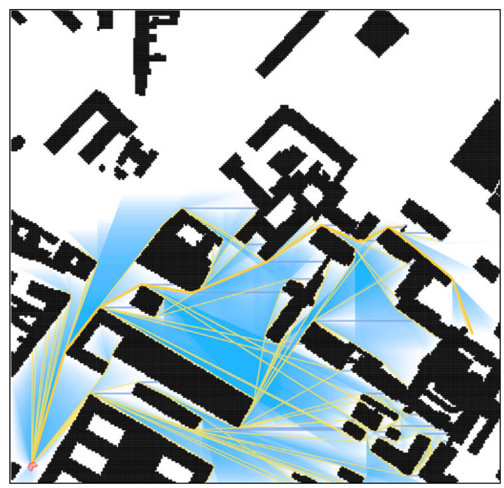
Level 5 is not presented in Fig. 5, because the transparency elements regarding the search process should be displayed by a controllable animation. Animation, as a dynamic visualization technique, has been shown to be helpful in explaining algorithmic behaviors and improving human understanding (Urquiza-Fuentes and Velázquez-Iturbide, 2009; Aysolmaz and Reijers, 2021). According to the cognitive load



(a) Grid world



(b) AR0602SR (Theta*)



(c) Berlin_0_256 (Anya)

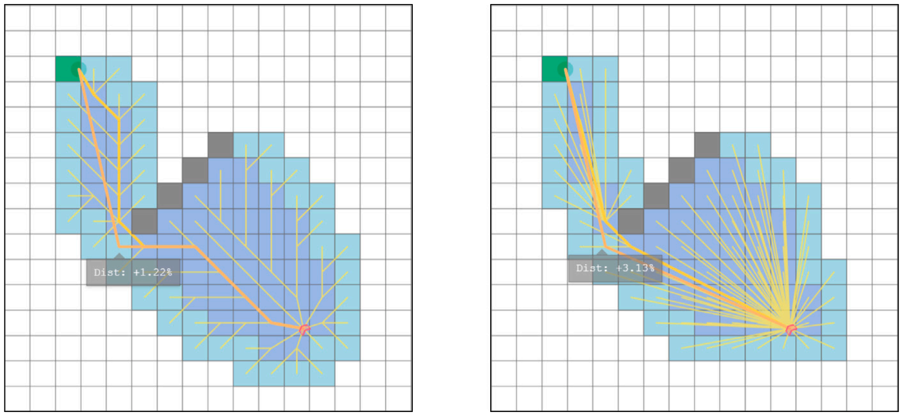
Fig. 6. Pathfinding visualizer, showing the grid world and two benchmark scenarios. Demo accessible here: http://dronectr.tudelft.nl/, ID: pathfinder.

theory (Sweller, 2011), the capacity of humans to process novel information is limited. Animation could make hidden dependencies among different components (e.g., nodes and branches) in a model more salient and thus reduce the cognitive load associated with comprehension (Aysolmaz and Reijers, 2021). Animation is quite suitable for presenting time-related procedural knowledge (Hundhausen et al., 2002; Aysolmaz and Reijers, 2021; Tversky et al., 2002). Through animation, the dynamic expansion, rewiring, and pruning of search trees can be visually observed. Moreover, animation should be both controllable and interactive given the complexity of the information it conveys (Tversky et al., 2002). Users should have the flexibility to view and review, stop and start the animation.

Fig. 5 actually hints at another reason for adopting the cumulative approach in classifying transparency levels: the transparency elements listed in Table 4 are not completely independent of one another. For example, the graph can be regarded as a discretized version of the search space, allowing one to roughly perceive the shape, size and boundary of the original search space. The search trees contain the location information of the explored nodes and are also shaped by the graph and search space. The search process can be viewed as a sequence of search trees, with the final tree representing the final result of the explored solutions (nodes and branches). Therefore, the new information introduced at deeper levels is actually dependent on the previous levels, but further elucidates the algorithm's inner workings. For an expert user already familiar with the algorithm, perhaps some information at lower levels can be hidden to avoid visual clutter without compromising comprehension.

5.2. Pathfinding visualizer

By implementing the approach proposed for information extraction and presentation, we developed a new web-based pathfinding visualizer[1] in JavaScript and OpenLayers (see Fig. 6). The visualizer shows the starting position of a generic vehicle (e.g., a drone, car or robot) in a grid world. When clicking on the vehicle icon, the final path (Level 0) and the target point will be presented. The remaining transparency elements and levels are implemented as five check boxes that follow the top-down structure. Users can activate the boxes to reveal more information about how the selected algorithm works. At the "Search Process" level, an interactive slider bar is implemented, allowing one to progressively step through the entire search process either forwards or backwards. Alternatively, the entire search process can be replayed forwards or backwards. Users can freely add or remove obstacles by blocking or unblocking grid elements after which the currently activated path-planning algorithm will generate a new,



(a) A* (b) Theta*

Fig. 7. The search trees and nodes of A* and Theta* (Level 4).

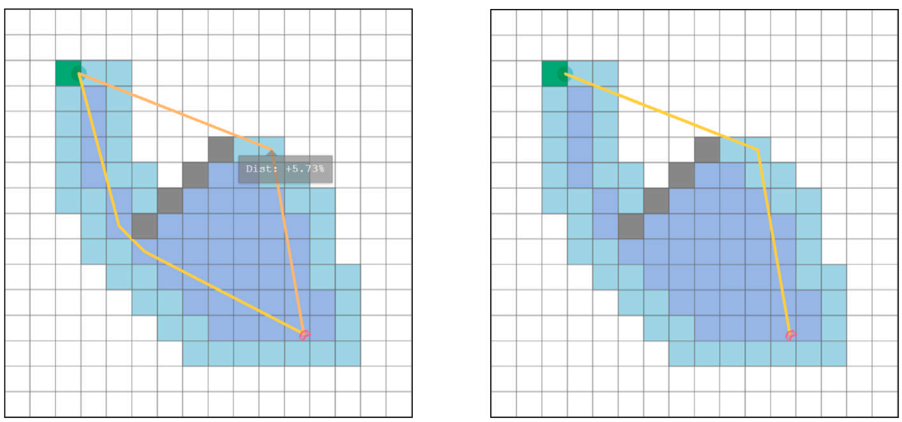


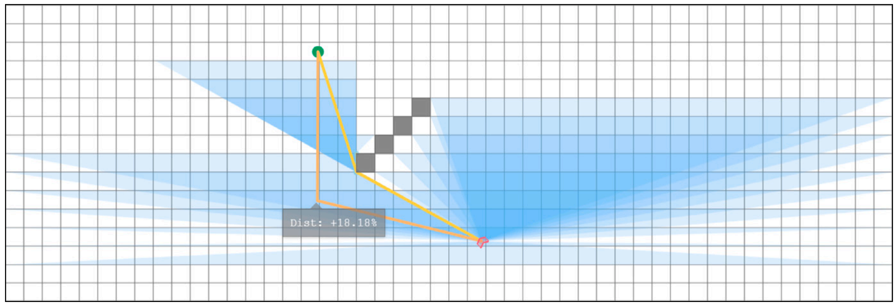
Fig. 8. Intervening in the Theta* algorithm.

collision-free path. Note that the grid is independent of the path-planning algorithms and only intended here to help users more easily create and remove obstacles. In addition, the start and target points in the visualizer need not necessarily be situated at the grid centers, which is more applicable to real-world scenarios. Over ten representative path-planning algorithms have been implemented in the visualizer. In this article, we select six of them to present briefly: A*, Theta*, Anya, Polyanya, Informed RRT* and BIT*. Other algorithms can be seen in our visualizer[1]. Please note that regardless of whether an algorithm is modern or not, as long as it relies on search trees, it can be visualized using our approach, such as Zeta*-SIPP (Zou and Borst, 2024). The nodes of the search trees do not necessarily have to be points; they can also take other shapes, as seen in Anya and Polyanya.

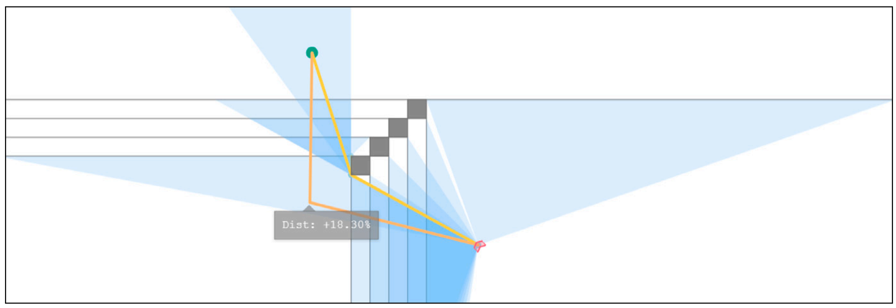
Fig. 7 shows the search trees and nodes of A* and Theta*. The light blue grids indicate the open nodes, while the dark blue (cornflower blue) grids denote the closed nodes. The yellow lines are the branches of search trees and the light orange thick line is the optimal path found by the algorithm. It can be clearly seen that A* is limited to 45-degree expansions while Theta* has a straighter and more directly oriented search tree, although their explored nodes are similar. Users can click on any node in the explored space to present an alternative path through the selected node and its corresponding cost. For example, when clicking on the nodes at the same position in Fig. 7, the distances in A* and Theta* will increase by 1.22% and 3.13% compared to their optimal paths, respectively. The alternative path consists of two parts: the explored path from the start to the selected node and the expected path from the selected node to the target. The explored path follows the branches of the search tree, whereas the expected path reflects the direction to the target. As shown in Fig. 8, users could also intervene in the output of the algorithm by choosing another (sub-optimal) path present in the stored search tree.

Anya and Polyanya are also A*-based algorithms. Unlike other algorithms, the search node of Anya and Polyanya is a tuple of a root (vertex) and an interval: (r, I), which can be viewed as a triangle as shown in Fig. 9. The explored space of Anya and Polyanya looks like a collection of rays emitted from a point source. In Anya, the rays radiate up or down row by row, while in Polyanya, the rays diverge polygon by polygon. The optimal path can be found when the target point is covered by the rays. Please note that in Anya, the combination of the triangles (rays) cannot be equivalent to the entire visible region. For example, as shown in Fig. 9(a), the area near the left and right boundaries cannot be fully covered by the rays from the start point. The branches of the search tree in Anya and Polyanya are generated by connecting the roots of the search nodes. The search node creates a continuous region that allows users to click on any point in the explored space to find an alternative path.

Informed RRT* and BIT* are sampling-based algorithms. They both use an *informed* search procedure to narrow the search space and speed up the convergence towards the optimal path. In Fig. 10, the sampling points are orange whereas the search nodes are green. They are near-uniformly distributed within the green ellipse, which represents the "current" search space and is defined by the "current" optimal path. A orange circle indicates the rewiring radius of Informed RRT*. Within the rewiring radius, the search tree is re-connected to generate straighter branches. Please note that Figs. 7–10 are provided for illustrative purposes only. We encourage the readers of this article to visit our web-based visualizer to better understand and interpret the information displayed in the figures.

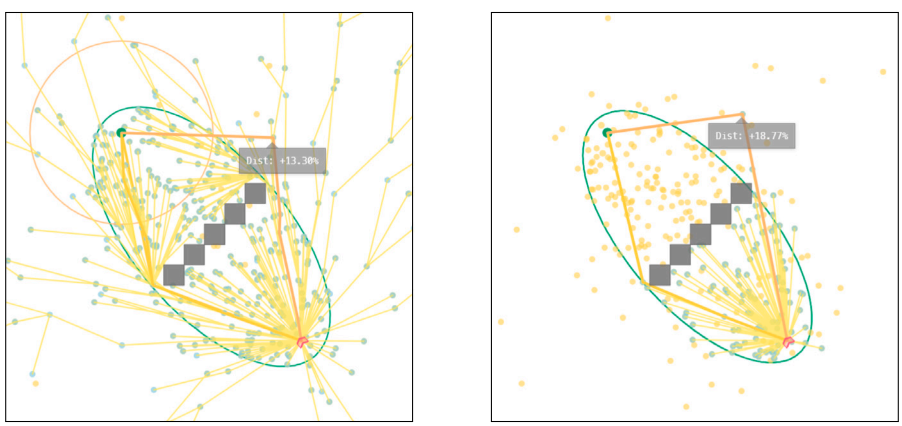


(a) Anya



(b) Polyanya

Fig. 9. The search trees and nodes of Anya and Polyanya (Level 4).



(a) Informed RRT* (b) BIT*

Fig. 10. The search trees and nodes of Informed RRT* and BIT* (Level 4).

6. User study

In the previous sections, we introduced methods of information extraction and presentation for achieving visual algorithmic transparency in path planning. In this section, we will present a user study evaluating the impact of algorithmic transparency on human understanding to validate its effectiveness. The goal of this experiment was to gain empirical insight into how transparency enables *non-experts* (in pathfinding algorithms) to correctly and confidently understand details of a pathfinding algorithm. The underlying motivation for this choice is the expectation that end users, system developers/engineers, and policy makers without a computer science background will eventually need to work with such algorithms in their professions.

6.1. Experiment setup

The experiments were conducted in our laboratory at TU Delft based on the pathfinding visualizer we developed, as shown in Fig. 11. As the laboratory is an enclosed room, only one participant was allowed at a time. The task for participants was to grasp the underlying path-planning algorithm through various levels of transparency. As shown in Fig. 12, in the experiments, the scenario remained consistent across all participants, algorithms and transparency levels, with fixed start and target points and modifiable obstacles. Participants were allowed to freely add or remove obstacles by blocking or unblocking grid elements. The experiments consist of two phases: training and measurement. The domain constraint considered in the measurement phase was that the path length cannot exceed a certain value. This constraint is very common when a vehicle needs to reach its destination within a limited time (e.g., battery power or emergency). Two different types of path-planning algorithms were chosen: Theta* and Informed RRT*. Each participant was assigned one of them for evaluation, following a between-participants design. During the experiments, participants were presented with different levels of transparency, starting from Level 1 to Level 5. At each level, participants were required to answer questions designed to test their understanding. Considering that maintaining focus for a long time is difficult, the entire experiment generally took 90 min, with the measurement phase having a maximum time limit

Fig. 11. Experiment setup in the laboratory.

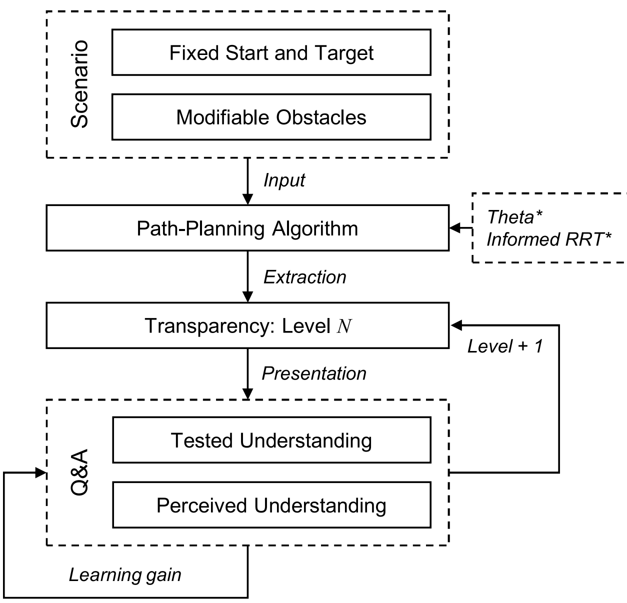


Fig. 12. Overview of experimental design.

of 60 min. Fig. 13 shows an example of the interface (Theta*, Level 5) presented to participants. Level 0 was excluded in the experiments according to three rounds of beta tests. During the beta tests, ambitious and exploration-minded participants found it difficult to complete the experiments containing Level 0 on time. Given that Level 1 only reflects the domain constraints imposed on the algorithms, the information it contains is relatively independent of algorithmic information. Hence, setting Level 1 as the baseline is also appropriate.

The questions to be answered included objective and subjective types, corresponding to tested and perceived understanding respectively. The objective questions were mostly multiple-choice, each with three incorrect choices and one correct choice. Each question included an "Unable to answer based on the available information" option, which participants were encouraged to select only if they felt they lacked sufficient information to answer accurately. The questions were devised based on the transparency elements in Table 4, covering from "Solution" to "Computational Process" categories. According to the DARPA's XAI definition (Gunning and Aha, 2019), human users should be able to understand the weaknesses of AI systems via XAI. Thus, we also asked participants about the disadvantage of the presented path-planning algorithm. When submitting an answer to an objective question, participants were asked to indicate their confidence in their responses (from 0 to 100). By considering both tested and perceived understanding, one's calibrated understanding could be assessed (McGuinness, 2004). Well-calibrated understanding indicates high confidence in correct answers and low confidence in incorrect ones. The final question was open-ended, designed to elicit participants' explanations regarding the inner workings of the algorithm. After all questions were submitted, the transparency level would automatically increase to the next level. At the next level, participants could then either adjust their initial answers or submit the same answers. This process continued until the end of the highest level of transparency. To reduce learning effects in the experiments, the ordering of the objective questions was randomized for each participant. The eight objective questions were as follows, with more details available in Appendix. These questions could also be answered through textual and/or verbal explanations, similar to those provided by textbooks and teachers. However, our focus was specifically on how visual transparency would enable participants to answer them.

Q1: The path-planning algorithm will always find the same path in the same situation/environment [True or False].
Q2: The path-planning algorithm will always find the true shortest path (if a path exists) [True or False].
Q3: What is the most accurate statement regarding the constraint for the path found by the path-planning algorithm?
Q4: What is the main advantage of the discretization method adopted by the path-planning algorithm?
Q5: What is the correct statement regarding the search nodes explored by the path-planning algorithm?
Q6: What is the correct statement regarding the search tree generated by the path-planning algorithm?
Q7: What strategy does the path-planning algorithm employ to attempt to find the shortest path?
Q8: What is the main disadvantage of the path-planning algorithm?

Before the experiment began, participants were required to undergo training to familiarize themselves with the interface elements, interactions, and questions along with their options. Dijkstra's algorithm (Dijkstra, 1959) was implemented and presented in this phase. The constraint and grid size were both different from those used in

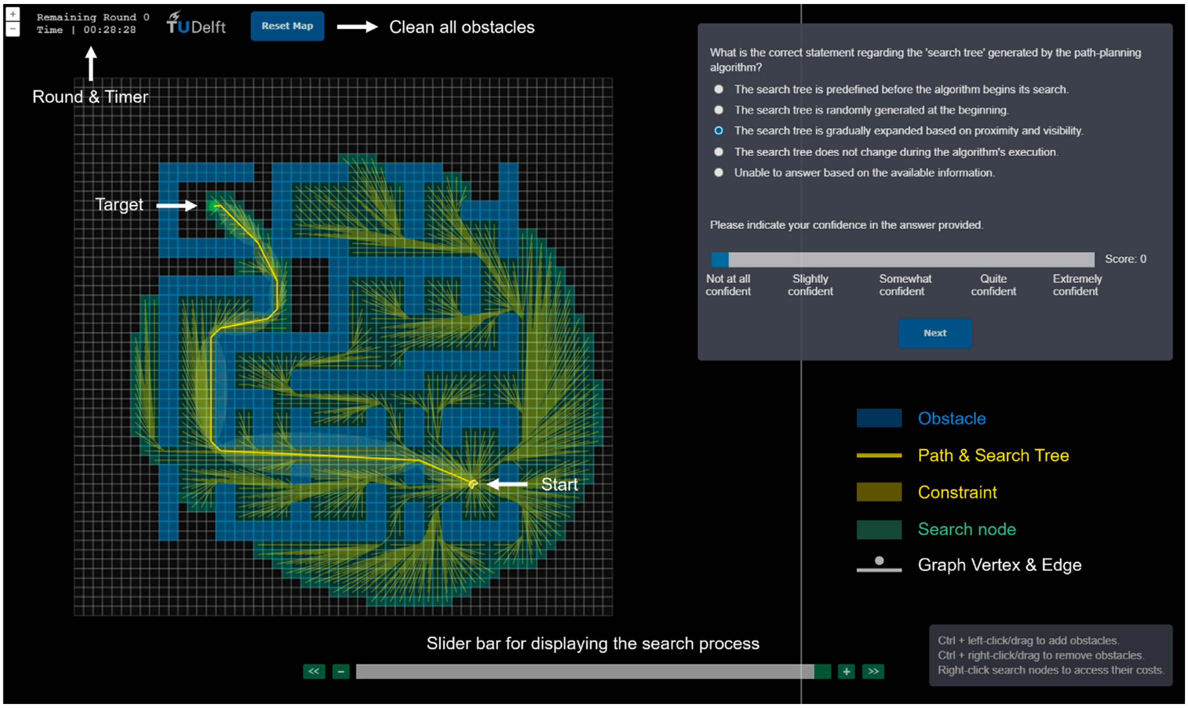


Fig. 13. Screenshot of the experiment interface (Theta*, Level 5). Demo accessible here: http://dronectr.tudelft.nl/, ID: understanding.

the measurement phase. For illustrative purposes, the transparency elements were presented in random combinations during training. Furthermore, participants were not informed of the presentation order of the transparency levels in the measurement phase. They only knew the remaining rounds they needed to complete. This setup could make it difficult for participants to predict what the supportive information would be available next, preventing them from intentionally skipping to higher levels to gather more information for answering questions. Participants were encouraged to answer the questions to the best of their ability using the information available to them. After the experiments, participants were asked to complete a post-hoc questionnaire in which they could indicate their preferences regarding the transparency elements using a five-point Likert scale.

6.2. Participants

Forty participants, all TU Delft staff and students from faculties Aerospace Engineering, Civil Engineering and Geosciences, Mechanical Engineering, and Technology, Policy and Management, volunteered to conduct the experiment. All of them had a general understanding of path-planning problems, but lacked in-depth knowledge of pathfinding algorithms. In fact, expert knowledge of pathfinding algorithms could bias or confound results, making it difficult to attribute understanding to the portrayed information. Participants were quasi-randomly assigned to the groups to create two balanced groups based on education level (see Fig. 14). This experiment was approved by the Human Research Ethics Committee (HREC) under number 4019.

6.3. Independent variables

The experiments had two main independent variables: (1) two path-planning algorithms (between-participants), and (2) five transparency levels (within-participants). Theta* and Informed RRT* were chosen because they are the classic (and still advanced) examples of graph and sampling based algorithms. They also have different implications for algorithm visualization, and thus potentially understanding. For each participant, the transparency levels all started at Level 1 and automatically increased to the next level after all questions were answered. Reverse or random ordering of the transparency levels was not adopted as a comparison group because the higher levels always encompass information from the lower ones, potentially influencing the learning outcomes of the lower levels. Moreover, the integration of information at the higher levels can circumvent the phenomenon of forgetting, ensuring that any enhanced understanding is attributable to the newly added information.

6.4. Dependent variables

Four types of dependent variables were considered: (1) understanding, (2) learning time, (3) interactions, and (4) preferences. Participant understanding was measured in terms of tested and perceived understanding (Mohseni et al., 2021; McGuinness, 2004; Lopes et al., 2022). The tested understanding represented the number of correct answers to the objective questions whereas the perceived understanding was measured by the participants' self-rated confidence. The learning time was recorded by the time it took participants to answer each question at each transparency level. To avoid learning effects in the experiments, the objective questions were presented in a random order. The interaction metrics include the number of algorithm executions and interactions with the map (e.g., grid activation and deactivation). Participants' preferences regarding the usefulness of the transparency elements (for the purpose of understanding) were measured on a five-point Likert scale.

6.5. Control variables

There were five main control variables: (1) interaction mode, (2) questions and options, (3) the order of transparency levels, (4) algorithm parameters, and (5) start and target points. At each transparency level, participants could freely interact with the path-planning algorithm by adding or removing obstacles, allowing them to observe how the solutions could be affected. This method can be regarded as *discovery learning* (Svinicki, 1998; Alfieri et al., 2011) or *active learning* (Michael, 2006), which emphasizes the role of engagement in learning and understanding (Hundhausen et al., 2002; Naps et al.,

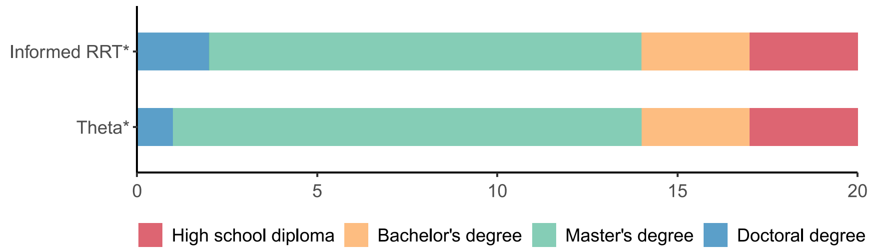


Fig. 14. Distribution of the participants' education levels.

Table 5
Means of dependent variables (DV) as a function of transparency level.

DV	Theta*					Informed RRT*				
	Level 1	Level 2	Level 3	Level 4	Level 5	Level 1	Level 2	Level 3	Level 4	Level 5
Hit ratio	0.32 (0.17)	0.42 (0.21)	0.44 (0.16)	0.49 (0.13)	0.54 (0.17)	0.47 (0.18)	0.53 (0.19)	0.57 (0.16)	0.61 (0.12)	0.63 (0.14)
Confidence	58.5 (13.8)	63.2 (17.6)	66.2 (17.1)	70.3 (17.8)	75.2 (18.3)	57.3 (13.8)	62.4 (13.3)	65.3 (13.8)	69.5 (14.7)	72.8 (16.6)
Learning time	774 (276)	526 (141)	552 (217)	491 (242)	474 (231)	769 (237)	612 (220)	560 (193)	531 (182)	674 (288)
Execution	63 (39)	39 (27)	40 (25)	28 (16)	22 (22)	109 (119)	69 (63)	74 (76)	63 (63)	34 (43)
Map interaction	257 (169)	140 (88)	123 (97)	68 (56)	77 (71)	244 (132)	145 (104)	137 (114)	110 (101)	92 (85)

Note: Standard deviations are in parenthesis.

2002; Grissom et al., 2003). In this manner, participants could maximize the use of available information to understand the algorithm, allowing us to more accurately assess the impact of transparency on human understanding. To maintain consistency, the scenario would be reset to its initial condition, removing all obstacles, before participants began answering questions at each level. Additionally, as indicated by Mayer (2004), guided discovery is generally more effective than pure discovery in learning. Therefore, the questions and options were integrated into the interface, rather than being added as a post-hoc quiz, to serve as guidance or learning objectives for participants, similar to an "open-book" exam. These questions and options remained consistent across all algorithms and levels of transparency. Only the correct answers to the questions may vary for different algorithms. As mentioned above, the order of transparency levels was fixed, ranging from Level 1 to Level 5. The algorithm parameters might also influence participants' comprehension as the algorithms' behaviors may differ depending on the parameters, such as the grid size in Theta* and the incremental distance in Informed RRT*. To facilitate comparison, the algorithm parameters, along with the start and target points, were kept fixed as well. For Theta*, the grid size is set to half the size of the obstacle grid. For Informed RRT*, the maximum number of iterations is set to 1000. To make the algorithm's behavior more apparent, the rewiring radius is set to $d_{st}/3$ and the incremental distance to $d_{st}/5$ where d_{st} is the distance between the start and target points.

6.6. Hypotheses

It was hypothesized that a higher transparency level would lead to a deeper understanding of path-planning algorithms, meaning higher fractions of correct answers and higher self-rated confidence scores. It was also expected that Informed RRT* would be comparatively more challenging to comprehend than Theta* due to the deterministic and structured nature of graph-based algorithms. This means that Theta* might be understood more quickly (less learning time) and effectively (higher learning scores) than Informed RRT* at the same transparency level. This hypothesis was driven by the observation that the behaviors of sampling-based path-planning algorithms are more difficult to predict because of their random sampling strategies.

6.7. Results

Table 5 presents the means and standard deviations of the dependent variables at different transparency levels for Theta* and Informed RRT*. In general, as the level of transparency increases, both the hit ratio and the confidence increase, while the learning time and the number of interactions (algorithm executions and interactions with the map) tend to decrease. Surprisingly, the hit ratio of Theta* is lower than that of Informed RRT*, contrary to our hypothesis. The learning time of Theta* is lower than that of Informed RRT* at the same level (except for Level 1), aligning with our expectations. The following sections will further analyze the results in more detail.

6.7.1. Data analysis and statistics

To compare Theta* and Informed RRT*, we first computed the means of dependent variables for each participant and then conducted Mann–Whitney U tests for statistical analysis. The effect size r was calculated in the Mann–Whitney U tests (small ≥ 0.1, medium ≥ 0.3, large ≥ 0.5), which is defined by the standardized test statistic z from the tests divided by the square root of the total number of observations. To compare different transparency levels in each algorithm group, we analyzed the dependent variables using Friedman tests, followed by Exact tests (Eisinga et al., 2017) with Bonferroni correction for further pairwise comparisons. Kendall's coefficient of concordance w was used to measure the effect size for the Friedman tests (small ≥ 0.1, medium ≥ 0.3, large ≥ 0.5). The significance level was set to 0.05. As the effect size reflects the magnitude of the difference between groups (Sullivan and Feinn, 2012), it is ideal to have both a statistically significant result ($p < 0.05$) and a large effect size to claim a clear and meaningful difference. A low effect size suggests that the statistical significance should be interpreted with caution.

6.7.2. Hit ratio

Fig. 15 shows the number of correct answers for each question at each transparency level. The bar plot above the heat map illustrates the cumulative number of correct answers for each question, whereas the bar plot on the right indicates the cumulative number of correct answers for each transparency level. It is evident that some questions are difficult to answer correctly, while others are not. The first question, Q1, asks whether the algorithm is deterministic or probabilistic. Most participants answered this question correctly across all transparency levels. However, several participants in Informed RRT* relied too much on their expectations of the algorithm and failed to notice that it generated different results in the same environment, leading to incorrect answers to Q1 at Level 1. Fortunately, when they observed the sampling points at Level 2, they correctly revised their answers. Q2 inquires whether the algorithm can find the true shortest

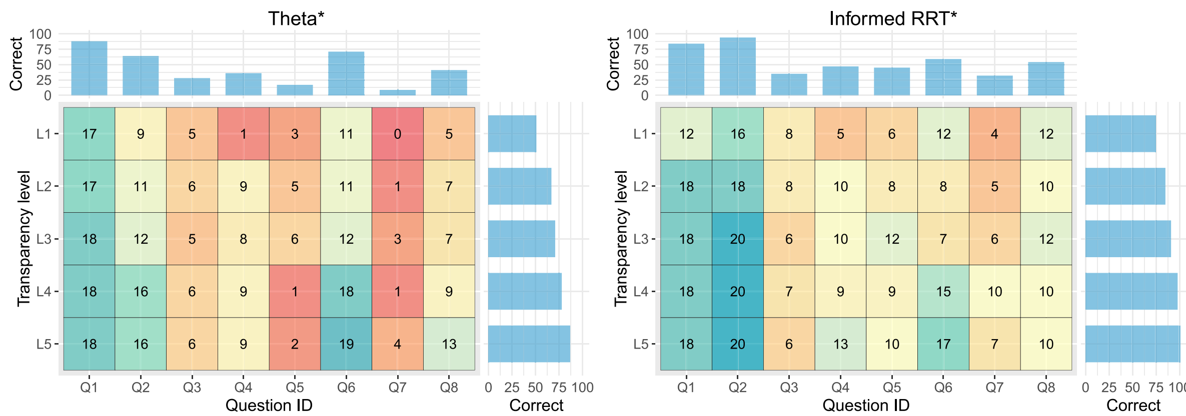


Fig. 15. The number of correct answers for each question at each transparency level.

path. As the path found by Theta* is close to the true shortest path, some participants failed to provide correct answers to this question, especially at lower transparency levels. Q3 asks what is constraining the path. The constraint implemented in the measurement phase was the maximum allowable path length. The results of Q3 suggest that the constraint should be more clearly indicated in the visualization. For example, a text box displaying "current path length/maximum allowable path length" can be attached. Q4 (discretization) and Q8 (overall) pertain to the advantages and disadvantages of the algorithms, which are difficult to answer because they cannot be directly perceived and need to be inferred from the visualization. Q5 is related to search nodes. There is an option in Q5 that misled many participants: the search node represents a visible region from a certain location. This option was designed originally for Anya and Polyanya. The search nodes of Theta* and Informed RRT* are not regions but points, making this option incorrect for both algorithms. However, the visualization cannot present points without a radius. For Theta*, the search nodes are represented by grid cells rather than points located at the grid centers. This misled participants, especially at Levels 4 and 5, causing them to change their correct answers to incorrect ones. Q6 is about search trees. The visualization of search trees (Level 4) positively impacts the accuracy of responses to Q6. Q7 asks about the strategy of the path-planning algorithm to find the shortest path (straighten the search tree). The results indicate that this question is difficult to correctly answer. Some participants suggested that at Level 5, the processes of expansion, rewiring, and pruning should all be presented, rather than just the final search tree of each step. Visualizing these intermediate sub-procedures at each step could help humans better understand the search process and strategy.

For further analysis, statistical tests were performed. Mann–Whitney U tests found a significant difference in the hit ratio between Theta* and Informed RRT* ($W = 94.5, p = 0.004, r = 0.707$). This result suggests that Informed RRT* may be easier to comprehend than Theta* through visualization, which contradicts our initial hypothesis. One possible reason is that, although the random sampling of Informed RRT* makes its behaviors difficult to predict, this randomness also makes its strategy and inner workings easy to observe. For example, by viewing the random sampling points generated by Informed RRT* (Level 2), participants may gain some insights into how the algorithm explores the space to find paths. In contrast, Theta* generates the same result given the same input. At lower levels of transparency, a few participants believed that the search tree of Theta* does not change during the algorithm's execution (Q6), and some participants assumed that Theta* considers the vertices of obstacles because it generates paths that appear as straight as possible (Q5 and Q7). Therefore, the algorithm type may affect human understanding of path-planning algorithms. For Theta*, a deeper level of transparency may be more necessary compared to Informed RRT* to help humans form a more accurate mental model.

Friedman tests revealed significant differences in the hit ratio between the transparency levels in both Theta* ($\chi^2(4) = 25.085$, p $< 0.001, w = 0.314$) and Informed RRT* ($\chi^2(4) = 16.446, p = 0.002, w = 0.206$). The effect sizes are not large, indicating that the substantive differences between transparency levels are not particularly pronounced. This is potentially because the number of questions is limited and some questions are too difficult to correctly answer (e.g., Q3, Q5 and Q7 in Fig. 15). Participants generally gained low hit ratios at each level (see Table 5). Additionally, the information was cumulatively presented as the level of transparency increased. The magnitude of differences between adjacent transparency levels is indeed not very large in terms of information amount (except for Level 5, search process). In Theta*, exact pairwise comparisons with Bonferroni correction further revealed significant differences in the hit ratio between Level 5 and Level 1 ($D = 41.5, p < 0.001$), Level 4 and Level 1 ($D = 30.0, p = 0.028$). In Informed RRT*, exact pairwise comparisons with Bonferroni correction further revealed a significant difference in the hit ratio between Level 5 and Level 1 ($D = 32, p = 0.013$). Compared to Level 1 (baseline), Level 4 (search tree) and Level 5 (search process) could offer significant insights into algorithms.

6.7.3. Confidence

Fig. 16 shows the distribution of confidence in being unable to answer, answering correctly, and answering incorrectly at each transparency level. Overall, the confidence tends to increase with the transparency level, regardless of whether the answer is correct or incorrect. This observation is in line with other research on transparency (Bhaskara et al., 2021) and decision-making (Tsai et al., 2008). Although the self-rated confidence is related to the amount of information received, the confidence in correct answers is generally higher than in incorrect answers, which also aligns with the findings of Daun et al. (2021). This indicates that the participants' self-rated confidence can indeed reflect their understanding to a certain degree. The confidence in being unable to answer decreases with increasing transparency, possibly because participants felt they should be able to answer as the amount of information increases, even if they still failed.

Friedman tests revealed significant differences in the confidence between the transparency levels in both Theta* ($\chi^2(4) = 50.394$, p $< 0.001, w = 0.630$) and Informed RRT* ($\chi^2(4) = 47.251$, p $< 0.001, w = 0.591$). In contrast to the hit ratio, the substantive differences in the confidence are evident. This is likely because confidence is independent of the number of questions and participants more easily became confident as they received more information (Tsai et al., 2008). Exact pairwise comparisons with Bonferroni correction further revealed differences between transparency levels, as shown in Table 6. The results indicate that Level 4 and Level 5 could significantly improve human perceived understanding.

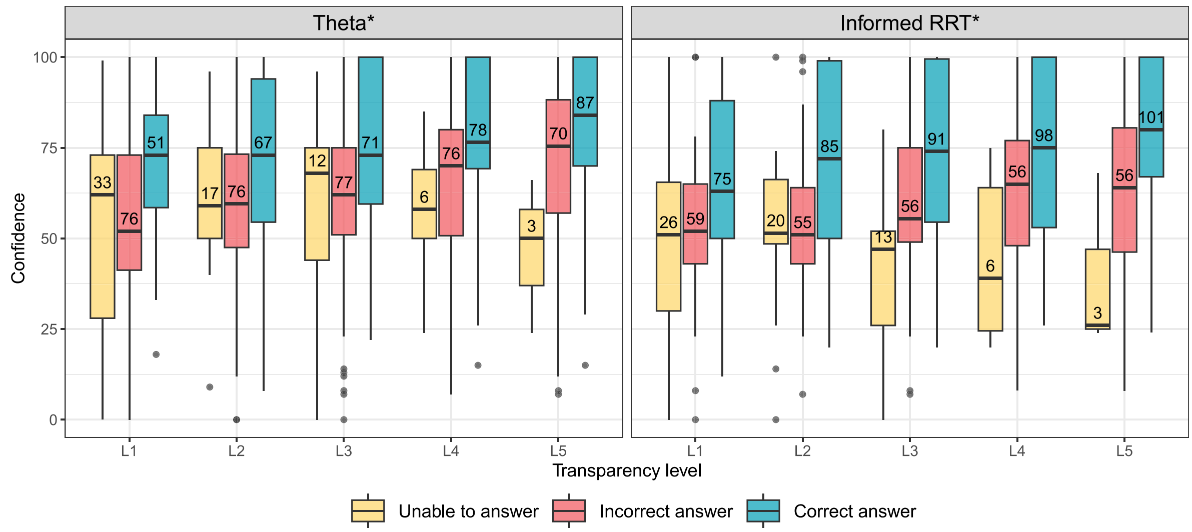


Fig. 16. Changes in confidence with increasing levels of transparency. The number above the median line indicates the number of observations in each box plot.

Table 6
Exact pairwise comparisons of confidence between transparency levels (with Bonferroni correction).

Transparency	Theta*					Informed RRT*				
	Level 1	Level 2	Level 3	Level 4	Level 5	Level 1	Level 2	Level 3	Level 4	Level 5
Level 1										
Level 2	1.000					0.308				
Level 3	0.990	1.000				0.028*	1.000			
Level 4	**	0.028*	0.308			**	0.028*	0.308		
Level 5	**	**	**	0.237		**	**	0.019*	1.000	

Note:
* $p < 0.05$.
** $p < 0.01$.

6.7.4. Calibrated understanding

Fig. 17 illustrates calibrated understanding based on the hit ratio and confidence, which reflect tested and perceived understanding, respectively. In this figure, the "unable to answer" responses were counted as incorrect answers with zero confidence, regardless of the original confidence ratings given by participants. This is because, when participants selected this option, they already knew their answers were incorrect (i.e., zero confidence in correctness). In contrast, when respondents selected other options, their ratings indicated a higher degree of confidence in the correctness of their answers. We expected that a good calibrated understanding would follow the diagonal line (from bottom left to top right) as the transparency level increases (McGuinness, 2004). Fig. 17 indicates that despite high variability in calibrated understanding among participants, the desired trend is generally evident in the mean (black lines). Compared to Informed RRT*, participants may exhibit overconfidence (i.e., a low hit ratio with high confidence) when learning about Theta*, indicating that the algorithm type may also play a vital role in human understanding.

6.7.5. Learning time

Fig. 18 presents the learning time distribution of Theta* and Informed RRT* at five transparency levels for the eight different questions. Overall, Theta* demonstrates lower learning time compared to Informed RRT*. As transparency levels increase, a downtrend in learning time is observed for both algorithms, suggesting that higher transparency may facilitate more efficient learning. However, the extent of this change varies by question. For Q7, Informed RRT* takes more time at Levels 4 and 5. This is likely because Levels 4 and 5 enable participants to observe the strategy of Informed RRT* in detail, but the processes of node expansion and tree rewiring are not easy to understand. For Q3, the constraint was not clearly stated, causing participants to spend much time exploring the algorithms to determine the exact parameter that was constraining the algorithms.

Mann–Whitney U tests found no significant difference between Theta* and Informed RRT* in terms of the total learning time. However, Table 5 suggests that participants may learn Theta* more quickly than Informed RRT*. This may be because Theta* is a deterministic, grid-based algorithm. It always generates the same result under the same conditions in the same environment and its graph (i.e., grid) remains fixed regardless of changes in obstacles. Each time a participant added or removed an obstacle, the visual information of Theta* did not change much. This apparently enables participants to become familiar with Theta* more quickly.

Friedman tests revealed significant differences in the learning time between the transparency levels in Theta* ($\chi^2(4) = 21.72$, $p < 0.001$, $w = 0.272$), but no significant difference in Informed RRT*. In Theta*, similar to the hit ratio, the effect size for the learning time is small. This may also be attributed to the correlations between transparency levels, which allowed participants to quickly grasp the newly added information. Moreover, the experiment had a maximum time limit, requiring participants to effectively manage their time for each level. Exact pairwise comparisons with Bonferroni correction further revealed significant differences in the learning time between Level 5 and Level 1 ($D = -43, p < 0.001$), Level 4 and Level 1 ($D = -36, p = 0.003$). Participants tended to spend more time on Level 1 because it was their first experience with the new algorithm. Participants spent more time checking Level 5 of Informed RRT* because understanding the random expansion of nodes and the rewiring of search trees is generally difficult.

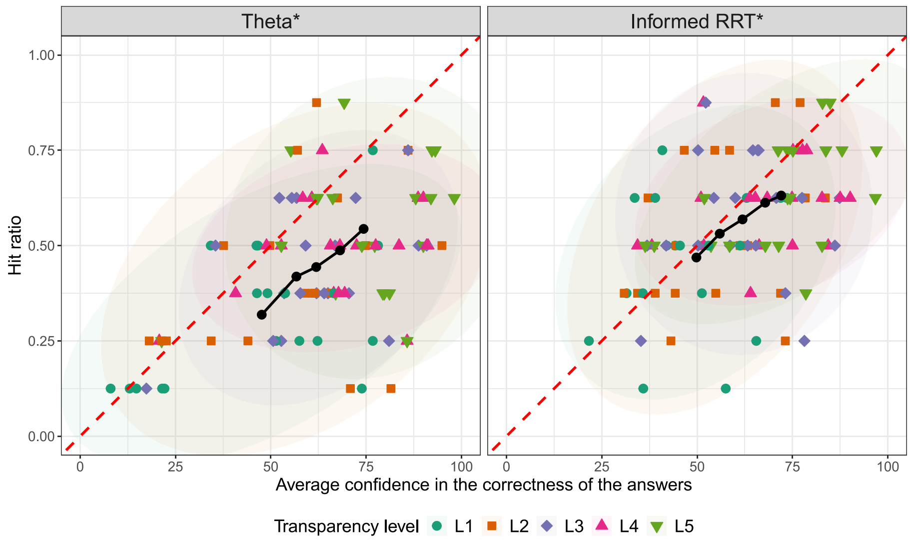


Fig. 17. Calibrated understanding by hit ratio and confidence, featuring 95% confidence ellipses and their centers for each transparency level.

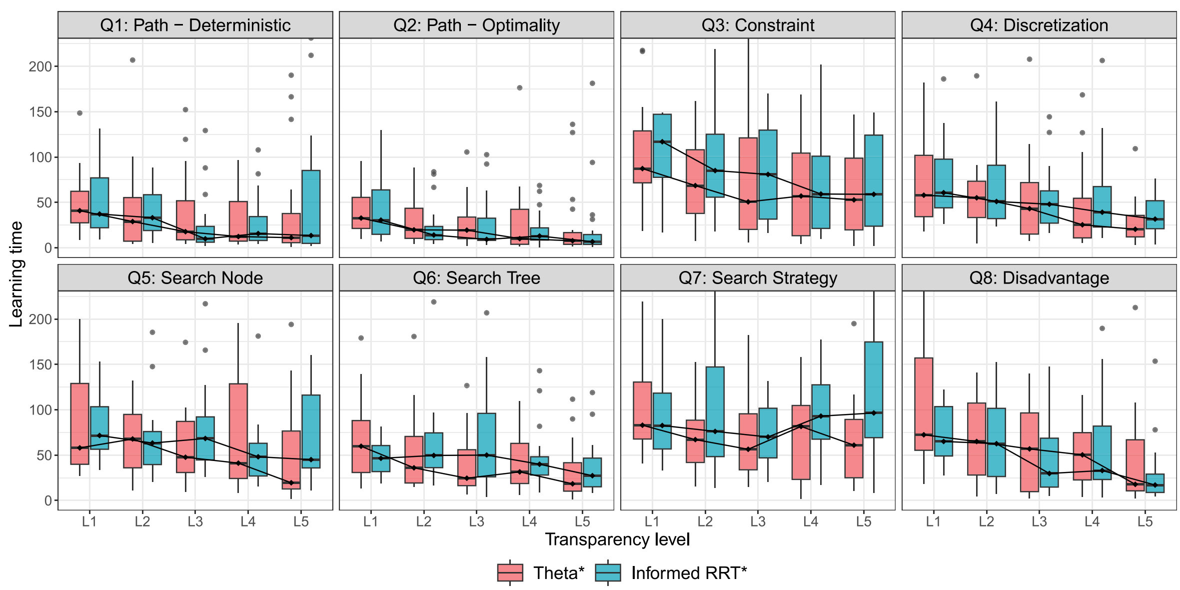


Fig. 18. Learning time for each question at each transparency level.

6.7.6. *Interaction*

There are two metrics for measuring interactions: the number of algorithm executions and the number of interactions with the map. Algorithm executions represent how many times participants challenged the algorithms by adding and removing obstacles that would lead to observable changes to their results while interactions with the map mainly refer to the number of obstacles added or removed. Considering that algorithm executions reflect the frequency with which participants see new information, we focus on this metric for analyzing interactions.

Fig. 19 presents the number of algorithm executions of Theta* and Informed RRT* for each question at each transparency level. The trend of this metric is similar to that of the learning time (see Fig. 18). In the figure, Q3 is the most prominent, as participants spent much effort creating various scenarios to examine the constraint. Informed RRT* generally exhibits higher median execution numbers and greater variability compared to Theta*. This is because Informed RRT* employs a random sampling strategy and some participants frequently executed Informed RRT* in the same scenario and observed changes in the results. Mann–Whitney U tests found no significant difference between Theta* and Informed RRT* in terms of the number of algorithm executions.

Friedman tests revealed significant differences in the number of algorithm executions between the transparency levels in both Theta* ($\chi^2(4) = 21.7$, $p < 0.001, w = 0.271$) and Informed RRT* ($\chi^2(4) = 19.719$, $p < 0.001, w = 0.246$). The effect sizes are both small, indicating that while differences in the interactions between transparency levels exist, they are not readily apparent. In addition to the relatively small differences between adjacent transparency levels, the fixed interaction mode and starting and target points (control variables) also reduce the need for a large number of interactions to achieve understanding. In Theta*, exact pairwise comparisons with Bonferroni correction further revealed significant differences in the algorithm executions between Level 5 and Level 1 ($D = -42, p < 0.001$), Level 4 and Level 1 ($D = -32.5, p = 0.009$). In Informed RRT*, exact pairwise comparisons with Bonferroni correction further revealed significant differences in the algorithm executions between Level 5 and Level 1

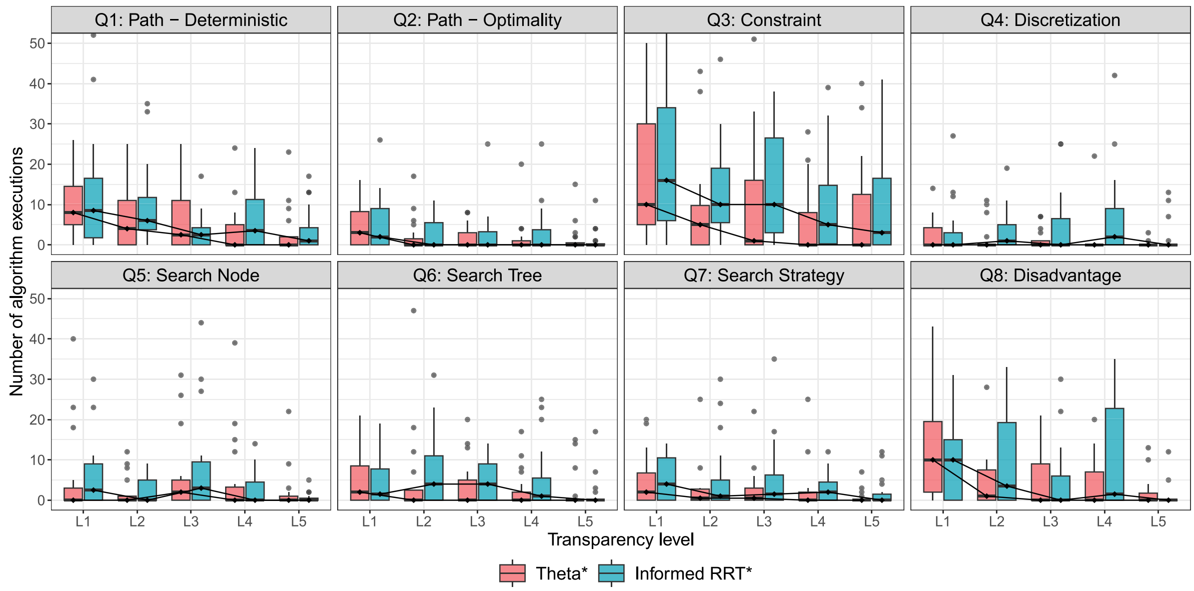


Fig. 19. The number of algorithm executions (reroutes) for each question at each transparency level.

($D = -42.5, p < 0.001$), Level 5 and Level 2 ($D = -28.5, p = 0.039$), Level 5 and Level 3 ($D = -28.5, p = 0.039$), Level 5 and Level 4 ($D = -30.5, p = 0.019$). This result can be linked to the learning time. The Spearman's rank correlation coefficient (0.625) indicates a strong correlation between the learning time and the number of algorithm executions. It seems that the greater the number of interactions, the longer the learning process takes. At Level 5 of Informed RRT*, the algorithm was executed less frequently, but the learning time was longer because participants spent considerable time examining the search process.

6.7.7. Preference

Fig. 20 illustrates the Likert scale ratings for the transparency elements presented in Table 4. The expansion, rewiring and pruning processes are integrated into the search process. Generally, except for the cost values, all other elements are considered highly useful for understanding both Theta* and Informed RRT*. This is probably because the tested algorithms both aim to find the shortest path and the cost value is intuitively reflected in the path length. However, for other types of costs in more real-life operational contexts, such as fuel consumption and risks, the cost values and their weights may become more useful. Compared to Informed RRT*, more participants found the search space and graph elements less useful for understanding Theta*. This may be because these elements can better help participants understand the random sampling strategy of Informed RRT*. However, for Theta*, they appear redundant due to the explored nodes and search trees.

7. Discussion

7.1. The impact of information extraction on algorithm speed

In our study, we have emphasized the important role that search trees play in the achievement of transparency and understanding. From the benchmark tests, we found that extracting search trees can significantly slow down the algorithms. The fewer the required search steps and branches, the shorter the time needed to extract the information. For example, although the line-of-sight checks in Theta* slow down the runtime, they can result in fewer branches, thereby speeding up the extraction process. Actually, the impact of information extraction depends not only on the number of branches but also on the difficulty of building a branch during path planning. A greater effort in constructing a branch generally suggests a lower ratio of time spent on branch extraction to branch construction. To generate shorter paths on grids, Theta* does not follow predefined edges but performs line-of-sight checks to create more direct paths between vertices. In this case, the effort required to build a branch is greater than that of directly following a predefined edge, thereby indirectly reducing the negative impact of information extraction on algorithm speed. When expanding a branch, RRT-based algorithms have to find the nearest node in the tree to the new sampling point in addition to line-of-sight checks, which further reduces the negative effect.

Although extracting search trees may be slow, fortunately, the data recording and extraction processes can be separated using Algorithm 4, allowing the latter to be performed in the background after finding paths. As shown in Fig. 2, the data recording process for Algorithm 4 slows down the original path-planning algorithms by less than 20%, with relatively limited impact on real-time performance of path planning. In practice, the search tree extraction process of Algorithm 4 can be executed in a separate thread to ensure that the main thread, which handles system interaction and path planning, remains unaffected. This can significantly mitigate the negative impact of information extraction on real-time operations.

In general, extracting information from algorithms to achieve transparency will slow down the algorithms anyway. For large-scale operations that demand real-time performance, achieving full transparency (Level 5) may be challenging. Users may experience delayed feedback from the algorithms, leading to slower responses to certain situations and perhaps even instability in closed-loop, human-automation interaction. In that case, lower levels of transparency may be more necessary. Compared to Level 5 (search process), Level 4 (search tree) only requires extracting the final search tree instead of extracting each tree at each step, largely reducing the negative impact of information extraction in real-time operations. For policy makers and algorithm designers, slower algorithm speed may be less of a concern, as they have sufficient time to audit an algorithm. In fact, a slower algorithm might even improve users' assessments (Park et al., 2019). The time spent waiting for the algorithm's output is often used to reflect on the problem at hand and rethink one's own solution, which helps prevent blindly following or dismissing the algorithm's results.

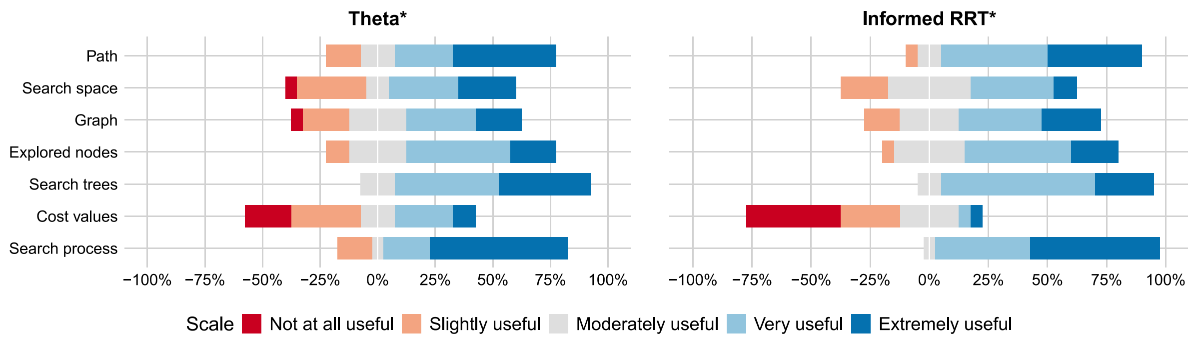


Fig. 20. Likert scale ratings for the transparency elements regarding the usefulness for understanding the path-planning algorithms.

7.2. Are sampling-based algorithms better than graph-based algorithms for understandability?

According to the results, it seems that sampling-based path-planning algorithms are more suitable for applications requiring algorithmic transparency, as the impact of information extraction is minimal and participants were generally able to better understand sampling-based algorithms than graph-based algorithms. Before conducting the user study, we assumed that Informed RRT* would be more difficult to understand than Theta*, due to its random sampling strategy and less organized visual presentation. However, the results indicated the opposite. This may be because the random sampling for path planning was more meaningful and in line with participants' expectations. Each time the algorithm was executed, Informed RRT* generated different random points, making it salient to participants that the algorithm used a random strategy to explore the space. For Theta*, although it is deterministic and highly structured, some participants mistakenly thought it also used a random strategy for exploration because its current node appeared to jump around randomly at each step. They did not understand why the current node behaved this way (recall that the current node is the node with the lowest f-value in the *open* list). For better understanding, it may be necessary to more clearly indicate the basis of the current node selection in the visualization. For example, provide additional textual and/or verbal explanations to clarify the rule, rather than requiring users to infer it from the visualization.

While the sampling-based algorithms showcased several advantages in this research related to understandability, they cannot replace graph-based algorithms. The main disadvantage of sampling-based algorithms is that they do not fully utilize prior knowledge of maps and they may sometimes get stuck in narrow passages (Tahirovic and Ferizbegovic, 2018; Wu et al., 2021). Stability and optimality are still the biggest obstacles for sampling-based algorithms in practice. Rather than drawing a rash conclusion ("better or not"), this research offers a new perspective for comparing algorithms on transparency and understanding criteria. These dimensions are also important to consider in real-world applications, especially in high-stakes domains where technology requires human oversight.

7.3. Is increased transparency always better?

The user study reveals that increased transparency generally leads to better understanding, as hypothesized. This result aligns with findings from other research (Springer and Whittaker, 2020; Cheng et al., 2019).

In the experiments, participants tended to rely on their expectations to formulate answers, especially when lacking sufficient information. The increased transparency can continuously correct their misconceptions and refine their mental models. However, this does not mean that they only changed incorrect answers to correct ones. Sometimes, when new information at subsequent levels violated participants' expectations, they felt more confused and provided more incorrect answers, resulting in a lower hit ratio. This phenomenon is similar to the mixed results identified in other research (Springer and Whittaker, 2020; Chen et al., 2023), suggesting that more information does not necessarily lead to better understanding and may instead result in greater confusion. This may occur when new information is insufficient to alleviate their confusion caused by the mismatch between their expectations and the truth, or the new information is unclear. For example, Theta* is an any-angle path-planning algorithm that ignores the edges of the graph. The mismatch between the paths and the graph (Level 2) made participants feel more confused. At lower levels of transparency, they initially thought that Theta* primarily explored the vertices of obstacles rather than grid by grid. This confusion also occurred when participants noticed that some search nodes and sampling points in Informed RRT* did not overlap. This inconsistency between the truth and human expectations indicates that higher levels of transparency are required to deepen understanding for some algorithms: *why it differs from what I expected*. It has been observed in the user study that participants' confusion was indeed alleviated (more confident and higher hit ratio) with higher levels of transparency. When the information is clear to users, they should eventually be able to gain a deeper understanding.

From the user study, we observed that participants' confidence increased with higher levels of transparency, regardless of whether their answers were correct or incorrect. This indicates that participants can become overconfident as transparency increases. This may be because participants misunderstood some transparency information, believing they understood it correctly when they actually did not. To alleviate this problem, one possible solution is to further increase transparency and provide more detailed, clear information. As learning deepens, users may identify their own mistakes. Another possible solution is to evaluate users' understanding and provide feedback to promptly correct their mental models. This idea is similar to the concept of model reconciliation (Sreedharan et al., 2021) as mentioned in the introduction, but estimating human mental models in real time remains a challenging issue. Fortunately, the procedures and strategies of algorithms are usually fixed, allowing users to learn them during training. In real-time operations, users may pay more attention to situational changes and algorithm behaviors. If users already understand the algorithm's strategy and basic principles, transparency information could help them quickly grasp the algorithm behaviors in a certain scenario, thereby improving their situation awareness (Endsley, 2023b, 1995).

The statistical analyses found significant differences in the dependent variables between transparency levels, but mostly with small effect sizes. This is largely because of the dependent relationships between transparency levels, where higher levels are built upon lower ones (see Fig. 5). As the level of transparency increases, the information regarding the inner workings of the algorithm is progressively and cumulatively revealed. If the information was chunked into more levels, the differences between transparency levels might be even less obvious. The results suggest that perhaps certain levels of transparency can be combined or removed to obtain potentially large effect sizes. For example, based on the benchmarking tests and user study findings, Level 4 (search tree) strikes a good balance between algorithm speed, human understanding and visual presentation complexity. Compared to Level 4, Levels 2 and 3 might be redundant and could be removed. Meanwhile, Levels 1 and 5 should be retained considering that Level 1 pertains to domain constraints (distinct type of information) and Level 5 to the search process (animation that contains a large amount of information). The elements "search space", "graph" and "explored nodes" can also be easily separated from Level 4 to further condense the transparency information. Since this research focused on the impact of domain-agnostic, algorithmic transparency on comprehension, future research could explore which transparency elements are preferred and used more frequently by operators (i.e. end users) during dynamic, real-time operations.

7.4. The importance of engagement in understanding and operations

As indicated by Hundhausen et al. (2002), Naps et al. (2002), Grissom et al. (2003) and Doherty and Doherty (2018), engagement plays an important role in enhancing human understanding. Although this research did not specifically examine the impact of engagement, we made efforts to maximize the participant engagement during the experiments (control variables). According to the engagement taxonomy in CS education (Naps et al., 2002), there are six categories for engagement with algorithm visualization: (1) No Viewing, (2) Viewing, (3) Responding, (4) Changing, (5) Constructing, and (6) Presenting. "No Viewing" refers to instruction without any visualization technology whereas "Viewing" is the core form of the visualization engagement. "Responding" involves answering questions concerning the visualization and "Changing" allows learners to change the algorithm's input to explore its behaviors. "Constructing" indicates learners build their own algorithm visualizations and "Presenting" suggests showing a visualization to an audience for feedback and discussion. This research incorporated "Viewing", "Responding" and "Changing" categories since our target audience was primarily non-experts and "Constructing" and "Presenting" were deemed less suitable.

For real-time operations, operators perform more effectively when their engagement level is high (Endsley, 2023b; Endsley and Kiris, 1995). As the level of automation increases, operators still need to be actively engaged into operational tasks and the control loop to understand what the system or AI is doing (Endsley, 2023a; Bainbridge, 1983). Otherwise, they could lose situation awareness and be unable to detect and address the system errors promptly. Empirical evidence has found a positive effect of transparency on situation awareness and operator performance due to the increased information (Bhaskara et al., 2020; van de Merwe et al., 2024; Stowers et al., 2020). In fact, transparency can also increase the engagement level of operators by offering ways to interact with the system. For example, through transparency, operators can visually examine the inner workings of the path-planning algorithm to understand why and how a specific path was proposed and/or transparency can support operators in 'what-if' probing (Wexler et al., 2020). They may actively check the transparency information, thus avoiding passively monitoring the system for extended periods and becoming bored.

7.5. Potential applications of algorithmic transparency

Algorithmic transparency in path planning holds numerous potential applications in the real world. For algorithm learners, transparency can facilitate a deeper understanding of path-planning algorithms, making complex concepts more accessible and less intimidating through visualization (Kehoe et al., 2001). For algorithm designers, it offers a valuable tool for visually debugging incorrect implementations and identifying opportunities for improving algorithms (Zheng et al., 2024). For example, Zeta*-SIPP was developed based on the inspiration drawn from the visualizations of Informed RRT* and BIT* (Zou and Borst, 2024). For system developers, transparency aids in evaluating the strengths and weaknesses of various algorithms, enabling the selection of the most suitable algorithm for a specific application domain. For operational users, transparency can enhance trust and acceptance of path-planning algorithms in operations, fostering productive human-AI collaboration (Endsley, 2023b). Although this article only explores the impact of transparency on human understanding, this understanding serves as the foundation for all these potential applications (Langer et al., 2021).

In safety-critical domains, such as air traffic control, algorithmic transparency in path planning also appears to be necessary for supporting human supervision (Zou and Borst, 2023). When automation fails to find a path, operators tend to seek more information to examine the underlying reasons behind the failure (Zou and Borst, 2023). The information overload issue also becomes more evident in tactical operations (Bhaskara et al., 2020). In high time-pressure situations, operators may even prefer a "black-box" approach (Hurter et al., 2022). In this case, a lower level of transparency may be more helpful. Operators may not have time to review the entire search process step by step, but they can quickly glance at the final search tree (Level 4) to understand what the algorithm explored. Our study also suggests that Level 4 can already facilitate a relatively clear understanding.

7.6. Limitations and future directions

This article summarizes the common steps of graph- and sampling-based path-planning algorithms, with a particular emphasis on search tree extraction for transparency. However, understanding how to construct graphs to preprocess the search space may also be important, especially for the algorithms based on navigation meshes (e.g., Polyanya) or random graphs (e.g., BIT*). Actually, for each transparency element, two different types of presentations can be defined: the output (static visualization) and the process (dynamic animation). For example, for explored nodes, one can present either the explored result or the exploration process of search nodes (without showing search trees). For the search space, one can either present it directly or illustrate the process of incorporating various domain constraints. This method may be helpful in understanding the individual elements in path planning.

From the user study, we learned that the visualization designed in this article needs further improvement. For example, more detailed information related to the search space and constraints should be provided, such as "current path length/maximum allowable path length". This would make it easier for users to check the relationship between the current results and the constraint bounds. For operational purposes, it is better to state the constraints in advance rather than letting users guess. In grid-based algorithms, if the search node is generated at the center of the grid rather than at the vertex, it may be helpful to mark this center point instead of the entire grid (or explain in advance). Marking the entire grid is a common approach to visualize the search node for grid-based algorithms, but it may mislead users into thinking that the algorithms explore the entire region within that grid (actually only the center will be explored). Furthermore, we portray the search tree at each step to reveal the search process (Level 5). However, the search process actually includes expansion, rewiring, and pruning, which can be separated for clearer presentation.

In addition to the visualization, textual and verbal explanations can also be useful in some scenarios. Textual explanations excel in providing precise, detailed descriptions that may be hard to capture in a visual format, such as abstract concepts and numerical data. In contrast, verbal explanations are inherently more human-like, fostering an interactive environment and deeper engagement for users. This research does not compare visual transparency with textual or verbal transparency because there are no readily available textual or verbal explanations for how path-planning algorithms work. To create them, we would need to extract explanations from the original papers that introduce these algorithms, such as Theta* (Daniel et al., 2010) and Informed RRT* (Gammell et al., 2014). Moreover, determining the appropriate level of detail to include in the textual or verbal explanations would also need to be addressed, which is beyond the scope of this article. For future research on transparent path planning, our work could serve as a reference point and a basis for comparison. The differences in effectiveness and user preferences between visual, textual, and verbal transparency could be further explored (Schmude et al., 2023).

The user study assessed the participants' understanding mainly through the eight questions. The questions were designed only to evaluate the understanding of the algorithm's inner workings. Future research could incorporate more questions to assess broader aspects of transparency and understanding, such as potential algorithm improvements and the environmental impact of solutions. Many participants reported that some questions were sometimes difficult to answer, which explains the relatively low average hit ratio. This may be because most participants were novices or beginners in path planning and were not familiar with its basic concepts. They needed to learn the relevant terms, quiz questions and options, and, most importantly, the algorithm itself within a limited time (20 min for training and 60 min for measurement). This task was indeed non-trivial for them.

Additionally, the experiment was intentionally designed in a generic setting, since path-planning algorithms typically follow fixed procedures, and revealing their inner workings does not require a real-world context. However, domain experts with extensive operational background may have better initial understanding of algorithms within their domain (Dikmen and Burns, 2022). Future studies should therefore also consider how domain knowledge might impact (the need for) transparency in understanding the working mechanisms of algorithms.

Finally, the transparency levels were implemented to be adaptable in our web-based pathfinding visualizer, enabling users to drive the interaction with the interface and access information on demand. This approach may avoid "automation surprises" from unexpected autonomy-driven changes in real-time operations (Calhoun et al., 2018). It is particularly suitable for human-centered domains such as air traffic control, where operators should maintain authority over and awareness of automation. Adaptive transparency that considers users' cognitive load and expertise levels could also be a promising area to explore. For example, by detecting users' confusion emotion (Arguel et al., 2017), the system could automatically adjust the level of transparency to provide more or less information and explanations. The ultimate goal of this research is to enhance human understanding through algorithmic transparency, rather than dynamically tailoring information delivery based on users' real-time cognitive states. The exploration of adaptive transparency is beyond the scope of this article. All in all, whether transparency is adaptive or adaptable, the findings of this research could provide a reference for both.

8. Conclusion

This article introduces a visual approach to algorithmic transparency in path planning, revealing the internal processes of path-planning algorithms through search trees. It summarizes the common steps in graph- and sampling-based path-planning algorithms and proposes methods for extracting and portraying the transparency information. Benchmark tests indicate that extracting all search trees during the search can significantly slow down the algorithms. To mitigate this, data recording and extraction can be separated, with search trees being extracted only when necessary. For portraying the extracted information, a new web-based pathfinding visualizer was developed based on JavaScript and OpenLayers, with over ten path-planning algorithms implemented. Six levels of transparency were designed and a user study was conducted to explore the impact of algorithmic transparency on human understanding. The results demonstrate that visual transparency allows inexperienced people to correctly and confidently understand the details of a path-planning algorithm and increased transparency generally improves understanding. The user study also shows that a lack of prior experience and knowledge of path planning sometimes can lead to confusion at higher levels of transparency, especially when the algorithm behaves contrary to expectations. Training, combined with a clear operational context and an algorithm that matches human expectations, could alleviate the need for greater transparency.

CRediT authorship contribution statement

Yiyuan Zou: Writing – original draft, Visualization, Validation, Software, Methodology, Investigation, Funding acquisition, Data curation, Conceptualization. **Clark Borst:** Writing – review & editing, Validation, Supervision, Resources, Project administration, Methodology, Funding acquisition, Conceptualization.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

The authors would like to thank the financial support from the China Scholarship Council (CSC) No. 202106830036 and all volunteers for their participation in this study.

Appendix. Quiz questions and options

The appendix presents the questions and corresponding options used in the experiments to test participants' understanding of path-planning algorithms. Each question aligns with a transparency element, covering different aspects of transparency information that users need to know. Comments on our thought process while designing these questions and our observations from the experiments are also attached to each question. We hope these can provide some reference for future research.

Q1: The path-planning algorithm will always find the same path in the same situation/environment.

A. True.
B. False.
C. Unable to answer based on the available information.

Comments:

This question was designed based on the "path" element to test whether participants can correctly understand the algorithm's nature (deterministic vs. probabilistic). It might be easier to answer when the algorithm is probabilistic since providing a counterexample is simpler than proving it. However, several participants still failed to answer correctly for Informed RRT*, especially at lower levels of transparency, due to incorrect expectations about the algorithm and a lack of in-depth exploration of it.

Q2: The path-planning algorithm will always find the 'true shortest' path (if a path exists).

A. True.
B. False.
C. Unable to answer based on the available information.

Comments:

This question was designed based on the "path" element to test whether participants can correctly understand the algorithm's optimality. Due to the limited rewiring radius and the restricted number of sampling points in the experiments, participants might easily recognize that the path found by Informed RRT* is not the true shortest. Compared to Informed RRT*, the path found by Theta* is deterministic and near-optimal, which might slightly increase the challenge for participants to answer this question correctly for Theta*.

Q3: What is the 'most accurate' statement regarding the constraint for the path found by the path-planning algorithm?

A. The path length cannot exceed a certain value.
B. The waypoints of the path cannot exceed the boundary of a fixed region.
C. The number of the waypoints cannot exceed a certain value.
D. The turning angle at the waypoint cannot exceed a certain value.
E. Unable to answer based on the available information.

Comments:

This question was designed based on the "search space" element to test whether participants can correctly understand the constraint via visualization. The constraint implemented in the measurement phase was the first option: maximum path length. The difficulty of this question lies in the fact that if the first option is correct, then the second option is correct, and the third option is also not entirely wrong. This is because if the path length is limited, there must exist an elliptical region to constrain the entire search space (recall the "informed" procedure in Informed RRT*). As the number of waypoints increases, the path length will also increase according to the triangle inequality theorem.

Q4: What is the main advantage of the 'discretization' method adopted by the path-planning algorithm?

A. Being able to represent the search space consistently and evenly regardless of changes in obstacles.
B. Being able to build direct connections between the corners/vertices of obstacles before the algorithm begins its search.
C. Being able to represent the search space exactly using relatively few polygons, especially in open areas with few obstacles.
D. Being able to discretize the search space progressively and continuously while conducting the search, instead of using a fixed graph.
E. Unable to answer based on the available information.

Comments:

This question was designed based on the "graph" element to test whether participants can perceive the advantage of the discretization method used in the algorithm. The first option is for regular grids, the second is for visibility graphs, the third is for navigation meshes, and the fourth is for sampling-based algorithms.

Q5: What is the correct statement regarding the 'search nodes' explored by the path-planning algorithm?

A. The search node is created based on the vertex (or center) of a predefined graph.
B. The search node is randomly generated based on a predefined sampling strategy.
C. The search node is generated at the corner/vertex of an obstacle.
D. The search node represents a visible region from a certain location.
E. Unable to answer based on the available information.

Comments:

This question was designed based on the "explored node" element to test whether participants can correctly understand the concept of search nodes. The first option is for graph-based or grid-based algorithms, the second is for sampling-based algorithms, the third is for visibility-graph-based algorithms, and the fourth is for Anya and Polyanya. However, participants tended to select the fourth option, particularly after viewing the search trees. This tendency arose probably because a node is visible from its parent node (a certain location) and some participants failed to recognize that the node of the tested algorithm represents a point rather than a region. Additionally, the node of Theta* is presented as a grid cell instead of a point located at the grid center, which may mislead participants about the algorithm.

Q6: What is the correct statement regarding the 'search tree' generated by the path-planning algorithm?

A. The search tree is predefined before the algorithm begins its search.
B. The search tree is randomly generated at the beginning.
C. The search tree is gradually expanded based on proximity and visibility.
D. The search tree does not change during the algorithm's execution.
E. Unable to answer based on the available information.

Comments:

This question was designed based on the "search tree" element to test whether participants can correctly understand the concept of search trees. The first option was to test whether participants can differentiate between predefined graphs and search trees. A search tree is the result of a search. The second option was designed as a trap, based on the fact that many algorithms in other fields generate a random initial solution as a starting point and then continuously optimize it. Also, the word "randomly" may mislead participants to associate this option with Informed RRT*. The fourth option was also designed as a trap. The deterministic algorithm (e.g., Theta*) consistently generates the same results and search trees, which might lead participants to believe that the fourth option is correct. Some participants were indeed misled by this.

Q7: What 'strategy' does the path-planning algorithm employ to attempt to find the shortest path?

A. Build direct visibility connections between the corners/vertices of obstacles by finding visible regions from explored locations.
B. Straighten the search tree by checking visibility from the current node's parent node to the current node's adjacent nodes.
C. Straighten the search tree by checking whether the current node could be a better parent of the nodes within its certain range.
D. Straighten the search tree by checking visibility between all explored nodes in the search space.
E. Unable to answer based on the available information.

Comments:

This question was designed based on the "search process" element to test whether participants can correctly understand the search strategy adopted by the algorithm. The first strategy is related to Anya and Polyanya, the second to Theta*, the third to RRT*-based algorithms and the fourth is borrowed from TO-AA-SIPP. This question is difficult to answer at lower levels of transparency.

Q8: What is the 'main disadvantage' of the path-planning algorithm?

A. Require significant preprocessing to discretize the search space before the algorithm begins its search, especially in complex environments (with many obstacles).

B. As the search space dimension increases (e.g., 3D, 4D, 5D, etc.), the time required to find a feasible path rises significantly (excluding the time for discretization).
C. Poor performance in environments with narrow passages, occasionally being unable to pass through them.
D. Turning angles at waypoints are limited to specific integer increments (e.g., if the increment is 15°, the turning angle can only be one of the following: 15°, 30°, 45°, and so on.).
E. Unable to answer based on the available information.

Comments:

This question was designed to test participants' overall understanding, specifically whether they can identify the algorithm's disadvantage via visualization. The first option pertains to generating irregular navigation meshes or visibility graphs, the second to grid-based path planning, the third to sampling-based algorithms, and the fourth is derived from the A* algorithm on grids (45-degree increments).

Data availability

We have shared the link to our visualization in the article.

References

Adadi, A., Berrada, M., 2018. Peeking inside the black-box: A survey on explainable artificial intelligence (XAI). IEEE Access 6, 52138–52160. http://dx.doi.org/10.1109/ACCESS.2018.2870052.

Alfieri, L., Brooks, P.J., Aldrich, N.J., Tenenbaum, H.R., 2011. Does discovery-based instruction enhance learning? J. Educ. Psychol. 103 (1), 1.

Almagor, S., Kottinger, J., Lahijanian, M., 2024. Temporal segmentation in multi agent path finding with applications to explainability. Artificial Intelligence 104087.

Almagor, S., Lahijanian, M., 2020. Explainable multi agent path finding. In: Proceedings of the 19th International Conference on Autonomous Agents and MultiAgent Systems. AAMAS '20, International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC, pp. 34–42.

Amershi, S., Weld, D., Vorvoreanu, M., Fourney, A., Nushi, B., Collisson, P., Suh, J., Iqbal, S., Bennett, P.N., Inkpen, K., Teevan, J., Kikin-Gil, R., Horvitz, E., 2019. Guidelines for human-AI interaction. In: Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems. CHI '19, Association for Computing Machinery, New York, NY, USA, pp. 1–13. http://dx.doi.org/10.1145/3290605.3300233, URL https://doi-org.tudelft.idm.oclc.org/10.1145/3290605.3300233.

Arguel, A., Lockyer, L., Lipp, O.V., Lodge, J.M., Kennedy, G., 2017. Inside out: detecting learners' confusion to improve interactive digital learning environments. J. Educ. Comput. Res. 55 (4), 526–551.

Aysolmaz, B., Reijers, H.A., 2021. Animation as a dynamic visualization technique for improving process model comprehension. Inf. Manag. 58 (5), 103478. http://dx.doi.org/10.1016/j.im.2021.103478, URL https://www.sciencedirect.com/science/article/pii/S0378720621000525.

Bainbridge, L., 1983. Ironies of automation. Automatica 19 (6), 775–779. http://dx.doi.org/10.1016/0005-1098(83)90046-8, URL https://www.sciencedirect.com/science/article/pii/0005109883900468.

Barredo Arrieta, A., Díaz-Rodríguez, N., Del Ser, J., Bennetot, A., Tabik, S., Barbado, A., Garcia, S., Gil-Lopez, S., Molina, D., Benjamins, R., Chatila, R., Herrera, F., 2020. Explainable artificial intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI. Inf. Fusion 58, 82–115. http://dx.doi.org/10.1016/j.inffus.2019.12.012, URL https://www.sciencedirect.com/science/article/pii/S1566253519308103.

Bekier, M., Molesworth, C., Brett, R., Williamson, A.M., 2011. Why air traffic controllers accept or refuse automated technology. In: 16th International Symposium on Aviation Psychology. p. 615.

Bhaskara, A., Duong, L., Brooks, J., Li, R., McInerney, R., Skinner, M., Pongracic, H., Loft, S., 2021. Effect of automation transparency in the management of multiple unmanned vehicles. Appl. Ergon. 90, 103243.

Bhaskara, A., Skinner, M., Loft, S., 2020. Agent transparency: A review of current theory and evidence. IEEE Trans. Hum.- Mach. Syst. 50 (3), 215–224. http://dx.doi.org/10.1109/THMS.2020.2965529.

Billings, C.E., 1997. Aviation Automation: the Search for a Human-Centered Approach. Lawrence Erlbaum Associates, Mahwah, NJ.

Bitzer, T., Wiener, M., Cram, W.A., 2023. Algorithmic transparency: Concepts, antecedents, and consequences–a review and research framework. Commun. Assoc. Inf. Syst. 52 (1), 293–331.

Brandão, M., Canal, G., Krivić, S., Luff, P., Coles, A., 2021a. How experts explain motion planner output: a preliminary user-study to inform the design of explainable planners. In: 2021 30th IEEE International Conference on Robot & Human Interactive Communication. RO-MAN, pp. 299–306. http://dx.doi.org/10.1109/RO-MAN50785.2021.9515407.

Brandão, M., Canal, G., Krivić, S., Magazzeni, D., 2021b. Towards providing explanations for robot motion planning. In: 2021 IEEE International Conference on Robotics and Automation. ICRA, pp. 3927–3933. http://dx.doi.org/10.1109/ICRA48506.2021.9562003.

Brandão, M., Coles, A., Magazzeni, D., 2021c. Explaining path plan optimality: Fast explanation methods for navigation meshes using full and incremental inverse optimization. Proc. Int. Conf. Autom. Plan. Sched. 31 (1), 56–64. http://dx.doi.org/10.1609/icaps.v31i1.15947, URL https://ojs.aaai.org/index.php/ICAPS/article/view/15947.

Brandao, M., Mansouri, M., Mohammed, A., Luff, P., Coles, A., 2022. Explainability in multi-agent path/motion planning: User-study-driven taxonomy and requirements. In: International Conference on Autonomous Agents and Multiagent Systems. AAMAS '22, International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC, pp. 172–180.

Brandão, M., Setiawan, Y., 2022. 'Why not this MAPF plan instead?'contrastive map-based explanations for optimal MAPF. In: ICAPS 2022 Workshop on Explainable AI Planning. pp. 1–8.

Bunt, A., Lount, M., Lauzon, C., 2012. Are explanations always important? A study of deployed, low-cost intelligent interactive systems. In: Proceedings of the 2012 ACM International Conference on Intelligent User Interfaces. pp. 169–178.

Calhoun, G.L., Ruff, H.A., Behymer, K.J., Frost, E.M., 2018. Human-autonomy teaming interface design considerations for multi-unmanned vehicle control. Theor. Issues Ergon. Sci. 19 (3), 321–352.

Cano, J., Yang, Y., Bodin, B., Nagarajan, V., O'Boyle, M., 2018. Automatic parameter tuning of motion planning algorithms. In: 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems. IROS, IEEE, pp. 8103–8109.

Chakraborti, T., Sreedharan, S., Kambhampati, S., 2020. The emerging landscape of explainable automated planning & decision making. In: Bessiere, C. (Ed.), Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20. International Joint Conferences on Artificial Intelligence Organization, pp. 4803–4811. http://dx.doi.org/10.24963/ijcai.2020/669, Survey track.

Chen, C., Feng, S., Sharma, A., Tan, C., 2023. Machine explanations and human understanding. Trans. Mach. Learn. Res. URL https://openreview.net/forum?id=y4CGF1A8VG.

Chen, J.Y., Lakhmani, S.G., Stowers, K., Selkowitz, A.R., Wright, J.L., Barnes, M., 2018. Situation awareness-based agent transparency and human-autonomy teaming effectiveness. Theor. Issues Ergon. Sci. 19 (3), 259–282.

Chen, J.Y., Procci, K., Boyce, M., Wright, J., Garcia, A., Barnes, M., 2014. Situation Awareness-Based Agent Transparency. Report, Aberdeen Proving Ground, MD, U.S. Army Research Laboratory.

Cheng, H.-F., Wang, R., Zhang, Z., O'connell, F., Gray, T., Harper, F.M., Zhu, H., 2019. Explaining decision-making algorithms through UI: Strategies to help non-expert stakeholders. In: Proceedings of the 2019 Chi Conference on Human Factors in Computing Systems. pp. 1–12.

Cui, M.L., Harabor, D.D., Grastien, A., 2017. Compromise-free pathfinding on a navigation mesh. In: Proceedings of the 26th International Joint Conference on Artificial Intelligence. IJCAI '17, AAAI Press, pp. 496–502.

Daniel, K., Nash, A., Koenig, S., Felner, A., 2010. Theta*: Any-angle path planning on grids. J. Artif. Int. Res. 39 (1), 533–579.

Daun, M., Brings, J., Obe, P.A., Stenkova, V., 2021. Reliability of self-rated experience and confidence as predictors for students' performance in software engineering: Results from multiple controlled experiments on model comprehension with graduate and undergraduate students. Empir. Softw. Eng. 26 (4), 80.

Diakopoulos, N., Koliska, M., 2017. Algorithmic transparency in the news media. Digit. Journal. 5 (7), 809–828.

Dijkstra, E., 1959. A note on two problems in connexion with graphs.. Numer. Math. 1, 269–271.

Dikmen, M., Burns, C., 2022. The effects of domain knowledge on trust in explainable AI and task performance: A case of peer-to-peer lending. Int. J. Hum.-Comput. Stud. 102792. http://dx.doi.org/10.1016/j.ijhcs.2022.102792, URL https://www.sciencedirect.com/science/article/pii/S1071581922000210.

Doherty, K., Doherty, G., 2018. Engagement in HCI: conception, theory and measurement. ACM Comput. Surv. 51 (5), 1–39.

Doshi-Velez, F., Kim, B., 2017. Towards a rigorous science of interpretable machine learning. arXiv preprintdoi:arXiv:1702.08608.

Dzindolet, M.T., Peterson, S.A., Pomranky, R.A., Pierce, L.G., Beck, H.P., 2003. The role of trust in automation reliance. Int. J. Hum.-Comput. Stud. 58 (6), 697–718.

Eiband, M., Schneider, H., Bilandzic, M., Fazekas-Con, J., Haug, M., Hussmann, H., 2018. Bringing transparency design into practice. In: 23rd International Conference on Intelligent User Interfaces. IUI '18, Association for Computing Machinery, New York, NY, USA, pp. 211–223. http://dx.doi.org/10.1145/3172944.3172961.

Eisinga, R., Heskes, T., Pelzer, B., Te Grotenhuis, M., 2017. Exact p-values for pairwise comparison of friedman rank sums, with application to comparing classifiers. BMC Bioinformatics 18, 1–18.

Endsley, M.R., 1995. Toward a theory of situation awareness in dynamic systems. Hum. Factors 37 (1), 32–64. http://dx.doi.org/10.1518/001872095779049543.

Endsley, M.R., 2023a. Ironies of artificial intelligence. Ergonomics 66 (11), 1656–1668.

Endsley, M.R., 2023b. Supporting human-AI teams: Transparency, explainability, and situation awareness. Comput. Hum. Behav. 140, 107574.

Endsley, M.R., Kiris, E.O., 1995. The out-of-the-loop performance problem and level of control in automation. Hum. Factors 37 (2), 381–394.

Gammell, J.D., Barfoot, T.D., Srinivasa, S.S., 2020. Batch informed trees (BIT*): Informed asymptotically optimal anytime search. Int. J. Robot. Res. 39 (5), 543–567. http://dx.doi.org/10.1177/0278364919890396.

Gammell, J.D., Srinivasa, S.S., Barfoot, T.D., 2014. Informed RRT: Optimal sampling-based path planning focused via direct sampling of an admissible ellipsoidal heuristic. In: 2014 IEEE/RSJ International Conference on Intelligent Robots and Systems. pp. 2997–3004. http://dx.doi.org/10.1109/IROS.2014.6942976.

Gammell, J.D., Srinivasa, S.S., Barfoot, T.D., 2015. Batch informed trees (BIT): Sampling-based optimal planning via the heuristically guided search of implicit random geometric graphs. In: 2015 IEEE International Conference on Robotics and Automation. ICRA, pp. 3067–3074. http://dx.doi.org/10.1109/ICRA.2015.7139620.

Gilbert, E.N., 1961. Random plane networks. J. Soc. Ind. Appl. Math. 9 (4), 533–543.

Goodman, B., Flaxman, S., 2017. European union regulations on algorithmic decision-making and a "right to explanation". AI Mag. 38 (3), 50–57. http://dx.doi.org/10.1609/aimag.v38i3.2741, URL https://ojs.aaai.org/index.php/aimagazine/article/view/2741.

Grissom, S., McNally, M.F., Naps, T., 2003. Algorithm visualization in CS education: comparing levels of student engagement. In: Proceedings of the 2003 ACM Symposium on Software Visualization. pp. 87–94.

Gunning, D., Aha, D., 2019. DARPA's explainable artificial intelligence (XAI) program. AI Mag. 40 (2), 44–58. http://dx.doi.org/10.1609/aimag.v40i2.2850, URL https://ojs.aaai.org/index.php/aimagazine/article/view/2850.

Gurka, J.S., Citrin, W., 1996. Testing effectiveness of algorithm animation. In: Proceedings 1996 IEEE Symposium on Visual Languages. IEEE, pp. 182–189.

Harabor, D., Grastien, A., Öz, D., Aksakalli, V., 2016. Optimal any-angle pathfinding in practice. J. Artif. Int. Res. 56 (1), 89–118.

Hart, P.E., Nilsson, N.J., Raphael, B., 1968. A formal basis for the heuristic determination of minimum cost paths. IEEE Trans. Syst. Sci. Cybern. 4 (2), 100–107. http://dx.doi.org/10.1109/TSSC.1968.300136.

Hechenberger, R., Stuckey, P.J., Harabor, D., Le Bodic, P., Cheema, M.A., 2020. Online computation of euclidean shortest paths in two dimensions. Proc. Int. Conf. Autom. Plan. Sched. 30 (1), 134–142. http://dx.doi.org/10.1609/icaps.v30i1.6654, URL https://ojs.aaai.org/index.php/ICAPS/article/view/6654.

Hilburn, B., Westin, C., Borst, C., 2014. Will controllers accept a machine that thinks like they think? The role of strategic conformance in decision aiding automation. Air Traffic Control. Q. 22 (2), 115–136.

Hoffman, R.R., Klein, G., 2017. Explaining explanation, part 1: theoretical foundations. IEEE Intell. Syst. 32 (3), 68–73.

Hundhausen, C.D., Douglas, S.A., Stasko, J.T., 2002. A meta-study of algorithm visualization effectiveness. J. Vis. Lang. Comput. 13 (3), 259–290.

Hurter, C., Degas, A., Guibert, A., Durand, N., Ferreira, A., Cavagnetto, N., Islam, M.R., Barua, S., Ahmed, M.U., Begum, S., et al., 2022. Usage of more transparent and explainable conflict resolution algorithm: air traffic controller feedback. Transp. Res. Procedia 66, 270–278.

Jamieson, G.A., Skraaning, G., Joe, J., 2022. The B737 MAX 8 accidents as operational experiences with automation transparency. IEEE Trans. Hum.- Mach. Syst. 52, 794–797. http://dx.doi.org/10.1109/THMS.2022.3164774.

Javaid, A., 2023. Local planner visualization project. https://github.com/abdurj/Local-Planner-Visualization-Project, (Accessed 5 December 2023).

Juozapaitis, Z., Koul, A., Fern, A., Erwig, M., Doshi-Velez, F., 2019. Explainable reinforcement learning via reward decomposition. In: Proceedings At the International Joint Conference on Artificial Intelligence. a Workshop on Explainable Artificial Intelligence. pp. 1–7.

Karaman, S., Frazzoli, E., 2011. Sampling-based algorithms for optimal motion planning. Int. J. Robot. Res. 30 (7), 846–894. http://dx.doi.org/10.1177/0278364911406761, arXiv:https://doi.org/10.1177/0278364911406761.

Kavraki, L., Svestka, P., Latombe, J.-C., Overmars, M., 1996. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. IEEE Trans. Robot. Autom. 12 (4), 566–580. http://dx.doi.org/10.1109/70.508439.

Kehoe, C., Stasko, J., Taylor, A., 2001. Rethinking the evaluation of algorithm animations as learning aids: an observational study. Int. J. Hum.-Comput. Stud. 54 (2), 265–284.

Kizilcec, R.F., 2016. How much information? Effects of transparency on trust in an algorithmic interface. In: Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems. pp. 2390–2395.

Kocielnik, R., Amershi, S., Bennett, P.N., 2019. Will you accept an imperfect ai? exploring designs for adjusting end-user expectations of ai systems. In: Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems. pp. 1–14.

Kottinger, J., Almagor, S., Lahijanian, M., 2021. MAPS-X: Explainable multi-robot motion planning via segmentation. In: 2021 IEEE International Conference on Robotics and Automation. ICRA, pp. 7994–8000. http://dx.doi.org/10.1109/ICRA48506.2021.9561893.

Kuffner, J., LaValle, S., 2000. RRT-connect: An efficient approach to single-query path planning. In: Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No.00CH37065). Vol. 2, pp. 995–1001 vol.2. http://dx.doi.org/10.1109/ROBOT.2000.844730.

Langer, M., Oster, D., Speith, T., Hermanns, H., Kästner, L., Schmidt, E., Sesing, A., Baum, K., 2021. What do we want from explainable artificial intelligence (XAI)? – A stakeholder perspective on XAI and a conceptual model guiding interdisciplinary XAI research. Artificial Intelligence 296, 103473. http://dx.doi.org/10.1016/j.artint.2021.103473, URL https://www.sciencedirect.com/science/article/pii/S0004370221000242.

Lawande, S.R., Jasmine, G., Anbarasi, J., Izhar, L.I., 2022. A systematic review and analysis of intelligence-based pathfinding algorithms in the field of video games. Appl. Sci. 12 (11), http://dx.doi.org/10.3390/app12115499, URL https://www.mdpi.com/2076-3417/12/11/5499.

Lee, J.D., See, K.A., 2004. Trust in automation: Designing for appropriate reliance. Hum. Factors 46 (1), 50–80. http://dx.doi.org/10.1518/hfes.46.1.50_30392, PMID: 15151155.

Lim, B.Y., Dey, A.K., Avrahami, D., 2009. Why and why not explanations improve the intelligibility of context-aware intelligent systems. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems. pp. 2119–2128.

Lipton, Z.C., 2018. The mythos of model interpretability: In machine learning, the concept of interpretability is both important and slippery. Queue 16 (3), 31–57. http://dx.doi.org/10.1145/3236386.3241340.

Liu, Q., Brandão, M., 2024. Generating environment-based explanations of motion planner failure: Evolutionary and joint-optimization algorithms. In: 2024 IEEE International Conference on Robotics and Automation. ICRA, pp. 15263–15269.

Lombrozo, T., 2006. The structure and function of explanations. Trends Cogn. Sci. 10 (10), 464–470.

Lopes, P., Silva, E., Braga, C., Oliveira, T., Rosado, L., 2022. XAI systems evaluation: A review of human and computer-centred methods. Appl. Sci. 12 (19), 9423.

Lundberg, S.M., Erion, G., Chen, H., DeGrave, A., Prutkin, J.M., Nair, B., Katz, R., Himmelfarb, J., Bansal, N., Lee, S.-I., 2020. From local explanations to global understanding with explainable AI for trees. Nat. Mach. Intell. 2 (1), 56–67.

Lundberg, S.M., Lee, S.-I., 2017. A unified approach to interpreting model predictions. In: Proceedings of the 31st International Conference on Neural Information Processing Systems. NIPS '17, Curran Associates Inc., Red Hook, NY, USA, pp. 4768–4777.

Lyons, J.B., 2013. Being transparent about transparency: A model for human-robot interaction. In: 2013 AAAI Spring Symposium Series. pp. 48–53.

Mayer, R.E., 2004. Should there be a three-strikes rule against pure discovery learning? Am. Psychol. 59 (1), 14.

McGuinness, B., 2004. Quantitative analysis of situational awareness (QUASA): Applying signal detection theory to true/false probes and self-ratings. In: Command and Control Research and Technology Symposium June. pp. 15–17.

Michael, J., 2006. Where's the evidence that active learning works? Adv. Physiol. Ed..

Mihailescu, C., 2023. Pathfinding visualizer. https://github.com/clementmihailescu/Pathfinding-Visualizer, (Accessed 5 December 2023).

Miller, T., 2019. Explanation in artificial intelligence: Insights from the social sciences. Artificial Intelligence 267, 1–38. http://dx.doi.org/10.1016/j.artint.2018.07.007, URL https://www.sciencedirect.com/science/article/pii/S0004370218305988.

Misra, D., 2023. Pathfinding visualizer threejs. https://github.com/dhruvmisra/Pathfinding-Visualizer-ThreeJS, (Accessed 5 December 2023).

Mohseni, S., Zarei, N., Ragan, E.D., 2021. A multidisciplinary survey and framework for design and evaluation of explainable AI systems. ACM Trans. Interact. Intell. Syst. (TiiS) 11 (3–4), 1–45. http://dx.doi.org/10.1145/3387166.

Naps, T.L., Rößling, G., Almstrum, V., Dann, W., Fleischer, R., Hundhausen, C., Korhonen, A., Malmi, L., McNally, M., Rodger, S., et al., 2002. Exploring the role of visualization and engagement in computer science education. In: Working Group Reports from ITiCSE on Innovation and Technology in Computer Science Education. Association for Computing Machinery, pp. 131–152.

Nothdurft, F., Richter, F., Minker, W., 2014. Probabilistic human-computer trust handling. In: Proceedings of the 15th Annual Meeting of the Special Interest Group on Discourse and Dialogue. SIGDIAL, pp. 51–59.

Nourani, M., Kabir, S., Mohseni, S., Ragan, E.D., 2019. The effects of meaningful and meaningless explanations on trust and perceived system accuracy in intelligent systems. In: Proceedings of the AAAI Conference on Human Computation and Crowdsourcing. Vol. 7, pp. 97–105.

van Paassen, M.M., Borst, C., Ellerbroek, J., Mulder, M., Flach, J.M., 2018. Ecological interface design for vehicle locomotion control. IEEE Trans. Hum.- Mach. Syst. 48 (5), 541–555. http://dx.doi.org/10.1109/THMS.2018.2860601.

Park, J.S., Barber, R., Kirlik, A., Karahalios, K., 2019. A slow algorithm improves users' assessments of the algorithm's accuracy. Proc. the ACM Hum.- Comput. Interact. 3 (CSCW), 1–15.

Patel, A., 2023. Red blob games. https://www.redblobgames.com/, (Accessed 5 December 2023).

Penrose, M., 2003. Random Geometric Graphs. Oxford University Press, http://dx.doi.org/10.1093/acprof:oso/9780198506263.001.0001.

Phillips, M., Likhachev, M., 2011. SIPP: Safe interval path planning for dynamic environments. In: 2011 IEEE International Conference on Robotics and Automation. pp. 5628–5635. http://dx.doi.org/10.1109/ICRA.2011.5980306.

Rader, E., Cotter, K., Cho, J., 2018. Explanations as mechanisms for supporting algorithmic transparency. In: Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems. pp. 1–13.

Rao, A.S., George, M.P., 1995. BDI agents: From theory to practice. In: Proceedings of the First International Conference on Multi-Agent Systems. ICMAS-95, pp. 312–319, URL https://www.aaai.org/Papers/ICMAS/1995/ICMAS95-042.pdf.

Rasmussen, J., 1985. The role of hierarchical knowledge representation in decisionmaking and system management. IEEE Trans. Syst. Man Cybern. SMC-15 (2), 234–243. http://dx.doi.org/10.1109/TSMC.1985.6313353.

Rasmussen, J., Vicente, K.J., 1989. Coping with human errors through system design: implications for ecological interface design. Int. J. Man-Mach. Stud. 31 (5), 517–534. http://dx.doi.org/10.1016/0020-7373(89)90014-X, URL https://www.sciencedirect.com/science/article/pii/002073738990014X.

Ribeiro, M.T., Singh, S., Guestrin, C., 2016. "Why Should I Trust You?": Explaining the predictions of any classifier. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. KDD '16, Association for Computing Machinery, New York, NY, USA, pp. 1135–1144. http://dx.doi.org/10.1145/2939672.2939778.

Riveiro, M., Thill, S., 2021. "That's (not) the output I expected!" On the role of end user expectations in creating explanations of AI systems. Artificial Intelligence 298, 103507.

Rudin, C., 2019. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. Nat. Mach. Intell. 1 (5), 206–215. http://dx.doi.org/10.1038/s42256-019-0048-x.

Schmude, T., Koesten, L., Möller, T., Tschiatschek, S., 2023. On the impact of explanations on understanding of algorithmic decision-making. In: Proceedings of the 2023 ACM Conference on Fairness, Accountability, and Transparency. pp. 959–970.

SESAR, 2022. AI in ATM: Transparency, Explainability, Conformance, Situation Awareness and Trust. Tech. rep., SESAR Joint Undertaking, URL http://mahaloproject.eu/wp-content/uploads/2022/12/Sesar-white-paper.pdf.

Shaffer, C.A., Cooper, M.L., Alon, A.J.D., Akbar, M., Stewart, M., Ponce, S., Edwards, S.H., 2010. Algorithm visualization: The state of the field. ACM Trans. Comput. Educ. (TOCE) 10 (3), 1–22.

Shen, B., Cheema, M.A., Harabor, D.D., Stuckey, P.J., 2022. Fast optimal and bounded suboptimal euclidean pathfinding. Artificial Intelligence 302, 103624. http://dx.doi.org/10.1016/j.artint.2021.103624, URL https://www.sciencedirect.com/science/article/pii/S0004370221001752.

Springer, A., Whittaker, S., 2020. Progressive disclosure: When, why, and how do users want algorithmic transparency information? ACM Trans. Interact. Intell. Syst. 10 (4), 1–32. http://dx.doi.org/10.1145/3374218, URL https://dl.acm.org/doi/10.1145/3374218.

Sreedharan, S., Chakraborti, T., Kambhampati, S., 2021. Foundations of explanations as model reconciliation. Artificial Intelligence 301, 103558. http://dx.doi.org/10.1016/j.artint.2021.103558, URL https://www.sciencedirect.com/science/article/pii/S0004370221001090.

Sreedharan, S., Hernandez, A.O., Mishra, A.P., Kambhampati, S., 2019. Model-free model reconciliation. In: Proceedings of the 28th International Joint Conference on Artificial Intelligence. pp. 587–594.

Sreedharan, S., Srivastava, S., Kambhampati, S., 2018. Hierarchical expertise level modeling for user specific contrastive explanations. In: IJCAI. pp. 4829–4836.

Stowers, K., Kasdaglis, N., Rupp, M.A., Newton, O.B., Chen, J.Y., Barnes, M.J., 2020. The IMPACT of agent transparency on human performance. IEEE Trans. Hum.-Mach. Syst. 50 (3), 245–253.

Sturtevant, N., 2012. Benchmarks for grid-based pathfinding. Trans. Comput. Intell. AI Games 4 (2), 144–148, URL http://web.cs.du.edu/~sturtevant/papers/benchmarks.pdf.

Sturtevant, N.R., 2023. Moving AI lab: Single agent search. https://www.movingai.com/SAS/, (Accessed 5 December 2023).

Sullivan, G.M., Feinn, R., 2012. Using effect size—or why the P value is not enough. J. Grad. Med. Educ. 4 (3), 279–282.

Svinicki, M.D., 1998. A theoretical foundation for discovery learning. Adv. Physiol. Ed. 275 (6), S4.

Sweller, J., 2011. Cognitive load theory. In: Psychology of Learning and Motivation. Vol. 55, Elsevier, pp. 37–76.

Tabrez, A., Luebbers, M.B., Hayes, B., 2020. A survey of mental modeling techniques in human–robot teaming. Curr. Robot. Rep. 1, 259–267.

Tahirovic, A., Ferizbegovic, M., 2018. Rapidly-exploring random vines (RRV) for motion planning in configuration spaces with narrow passages. In: 2018 IEEE International Conference on Robotics and Automation. ICRA, IEEE, pp. 7055–7062.

Toma, A.-I., Hsueh, H.-Y., Jaafar, H.A., Murai, R., Kelly, P.H., Saeedi, S., 2021. PathBench: A benchmarking platform for classical and learned path planning algorithms. In: 2021 18th Conference on Robots and Vision. CRV, pp. 79–86. http://dx.doi.org/10.1109/CRV52889.2021.00019.

Tsai, C.I., Klayman, J., Hastie, R., 2008. Effects of amount of information on judgment accuracy and confidence. Organ. Behav. Hum. Decis. Process. 107 (2), 97–105.

Tversky, B., Morrison, J.B., Betrancourt, M., 2002. Animation: can it facilitate? Int. J. Hum.-Comput. Stud. 57 (4), 247–262.

Urquiza-Fuentes, J., Velázquez-Iturbide, J.Á., 2009. A survey of successful evaluations of program visualization and algorithm animation systems. ACM Trans. Comput. Educ. (TOCE) 9 (2), 1–21.

van de Merwe, K., Mallam, S., Nazir, S., 2024. Agent transparency, situation awareness, mental workload, and operator performance: A systematic literature review. Hum. Factors 66 (1), 180–208.

van der Waa, J., Nieuwburg, E., Cremers, A., Neerincx, M., 2021. Evaluating XAI: A comparison of rule-based and example-based explanations. Artificial Intelligence 291, 103404. http://dx.doi.org/10.1016/j.artint.2020.103404.

Vicente, K., Rasmussen, J., 1992. Ecological interface design: theoretical foundations. IEEE Trans. Syst. Man Cybern. 22 (4), 589–606. http://dx.doi.org/10.1109/21.156574.

Vilone, G., Longo, L., 2021. Notions of explainability and evaluation approaches for explainable artificial intelligence. Inf. Fusion 76, 89–106.

Wang, D., Yang, Q., Abdul, A., Lim, B.Y., 2019. Designing theory-driven user-centric explainable AI. In: Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems. CHI '19, Association for Computing Machinery, New York, NY, USA, pp. 1–15. http://dx.doi.org/10.1145/3290605.3300831.

Wexler, J., Pushkarna, M., Bolukbasi, T., Wattenberg, M., Viegas, F., Wilson, J., 2020. The what-if tool: Interactive probing of machine learning models. IEEE Trans. Vis. Comput. Graphics 26 (01), 56–65. http://dx.doi.org/10.1109/TVCG.2019.2934619.

Wu, Z., Meng, Z., Zhao, W., Wu, Z., 2021. Fast-RRT: A RRT-based optimal path finding method. Appl. Sci. 11 (24), 11777.

Xu, X., 2023. PathFinding.js. https://github.com/qiao/PathFinding.js, (Accessed 5 December 2023).

Xue, F., Kumar, P.R., 2004. The number of neighbors needed for connectivity of wireless networks. Wirel. Netw. 10 (2), 169–181.

Yakovlev, K., Andreychuk, A., 2021. Towards time-optimal any-angle path planning with dynamic obstacles. Proc. Int. Conf. Autom. Plan. Sched. 31 (1), 405–414, URL https://ojs.aaai.org/index.php/ICAPS/article/view/15986.

Yap, P., Burch, N., Holte, R., Schaeffer, J., 2011. Block A*: Database-driven search with applications in any-angle path-planning. Proc. the AAAI Conf. Artif. Intell. 25 (1), 120–125. http://dx.doi.org/10.1609/aaai.v25i1.7813, URL https://ojs.aaai.org/index.php/AAAI/article/view/7813.

Zheng, K., Harabor, D., Wybrow, M., 2024. Posthoc: A visualisation framework for understanding search. In: ICAPS 2024 System's Demonstration Track. pp. 1–3, URL https://openreview.net/forum?id=VXHSST7CRi.

Zhu, J., Liapis, A., Risi, S., Bidarra, R., Youngblood, G.M., 2018. Explainable AI for designers: A human-centered perspective on mixed-initiative co-creation. In: 2018 IEEE Conference on Computational Intelligence and Games. CIG, pp. 1–8. http://dx.doi.org/10.1109/CIG.2018.8490433.

ZJU-FAST-Lab, 2023. C++ implementation and visualization of some sampling-based path planners. https://github.com/ZJU-FAST-Lab/sampling-based-path-finding, (Accessed 5 December 2023).

Zou, Y., Borst, C., 2023. Investigating transparency needs for supervising unmanned air traffic management systems. In: 13th SESAR Innovation Days. pp. 1–9.

Zou, Y., Borst, C., 2024. Zeta*-SIPP: Improved time-optimal any-angle safe-interval path planning. In: 33rd International Joint Conference on Artificial Intelligence. pp. 6823–6830.