Towards Explainable Road Navigation Systems

Khalid Alsheeb[1] and Martim Brandão[2]

***Abstract*— Road navigation systems are important systems for pedestrians, drivers, and autonomous vehicles. Routes provided by such systems can be unintuitive, and may not contribute to an improvement of users' mental models of maps and traffic. Automatically-generated explanations have the potential to solve these problems. Towards this goal, in this paper we propose algorithms for the generation of explanations for routes, based on properties of the road networks and traffic. We use a combination of inverse optimization and diverse shortest path algorithms to provide optimal explanations to questions of the type "why is path A fastest, rather than path B (which the user provides)?", and "why does the fastest path not go through waypoint W (which the user provides)?". The explanations reveal properties of the map—such as speed limits, congestion and road closure—that are not compatible with users' expectations, and the knowledge of which would make users prefer the system's path. We demonstrate the explanation algorithms on real map and traffic data, and conduct an evaluation of the properties of the algorithms.**

I. Introduction

Road navigation systems, which plan routes for driving, cycling, or walking paths, are widespread systems present in cars, smartphones, and autonomous vehicle software. Google Maps services, for example, had more than one billion monthly active users in 2020 [1]. Routes computed by such systems can often be unintuitive, particularly when users of navigation systems already have partial knowledge of the area they are navigating. This is because maps, road networks, speed limits, circulation rules and real-time traffic factors can interact in complex ways. The premise of this paper is that automatically generated explanations for routes may potentially help users understand navigation decisions, improve their mental models [2] of maps and traffic, and help calibrate their trust in navigation systems.

In this paper we propose algorithms that automatically generate explanations for questions of the type "why is path A fastest, rather than path B (which the user provides)?", and type "why does the fastest path not go through waypoint W (which the user provides)?". We leverage a combination of inverse optimization methods [2], [3] and diverse shortest path methods [4] to generate explanations of optimal length, i.e. that refer to a minimum possible amount of factors in the road network. Our explanations can justify the optimality of paths based on a variety of factors, from road closure and permitted traffic direction (one-way/two-way), to speed limits and congestion. The methods work by computing a minimal set of changes to road network parameters (i.e. congestion, permitted travel direction, temporary closure, speed limits) that lead an input path to become optimal. The waypoint explanation problem is more complex and requires finding the optimal explanation across all possible paths through the waypoint—for which we develop a new anytime asymptotically-optimal algorithm.

The contributions of this paper are thus the following:

1) We propose a method based on Inverse Shortest Paths to compute explanations for why a user-provided path is not optimal;
2) We propose an asymptotically-optimal anytime algorithm to compute the explanations for why a user-provided waypoint is not part of the optimal path;
3) We demonstrate the capabilities of the methods on real map and traffic data; and quantitatively evaluate the anytime algorithm against a sub-optimal approach.

II. Related Work

Recent studies have shed light into challenges and requirements for explainability in navigation systems, both in smartphone apps [5] and Autonomous Vehicles (AVs) [6], [7]. Most related to our paper, Chazette et al. [5] identified "why was my route chosen?" as a common understandability issue in navigation systems, and concluded that route explanations need to provide information about congestion, closures, and other factors—as we implemented in this paper. The study further showed that navigation systems with context-sensitive explanations lead to increased system use and satisfaction with routes. In AVs, Omeiza et al. [6] conducted a survey of explanations in autonomous driving in terms of motivations for their use, categories of explanations, datasets, challenges, and other issues. The survey described explainability as important in both planning and control of AVs, relating explanations in these contexts to factors of road networks, road signs, and road quality. Similarly to our paper, it identifies questions such as "why did you turn left?" as important for stakeholders. Motivated by cognitive and social studies of explanations [8], in this paper we focus specifically on *contrastive* explanations—that contrast the systems' plan to a user-specified plan or waypoint (e.g. answering "why did you turn left *instead of right*?", "why did you *not* take road W?").

Our paper is also related to efforts towards improving the user experience of traditional vehicle and pedestrian navigation systems, such as new interfaces that improve the acquisition of spatial knowledge [9] and systems that improve the communication and following of paths [10].

The methods we propose in this paper are related to the literature on inverse shortest paths [3] and their use for

[1]Khalid Alsheeb is with King's College London, London, UK. khalid.alsheeb@kcl.ac.uk
[2]Martim Brandão is with King's College London, London, UK. martim.brandao@kcl.ac.uk

path-finding explanations in robotics and computer game contexts [2], as well as for multi-agent path finding [11]. They are also aligned with much recent work on the topic of eXplainable AI Planning (XAIP) [12], [13], which focuses on task (instead of path) planning. Our explanation methods also relate to recent work on adversarial attacks on networks [14], where the goal is to compute a minimal set of edges to remove from a graph so as to make a desired path optimal.

Finally, our anytime method makes use of an adaptation of a diverse shortest path algorithm [4]. Diverse shortest paths algorithms [4], [15] compute a set of short paths between the same start and goal locations that are different from each other, and we use them here as a way to asymptotically obtain all paths that traverse a given waypoint.

III. Background

A. Multi-Directed Graph

A directed graph is a graph $G = (V, E)$, where E is the set of edges and V the set of nodes. Edges are defined in a specific direction, so if $u \neq v$ for $u, v \in V$, then $(u, v) \neq (v, u)$ for $(u, v), (v, u) \in E$. A multi-directed graph is a directed graph where a pair of nodes can have multiple edges between them in the same direction.

B. Shortest Path

A path is a sequence of edges that connect a set of nodes, starting from a start node and ending with a goal node. The shortest path problem can be formulated as a linear program:

$$\min_{\boldsymbol{x}} \quad \boldsymbol{w}^T \boldsymbol{x} \tag{1a}$$
$$\text{s.t.} \quad A\boldsymbol{x} = \boldsymbol{b}, \tag{1b}$$
$$\boldsymbol{x} \in \mathbb{R}_{0+}^{|V|}, \tag{1c}$$

where $x_j = 1$ if the edge e_j belongs to the shortest path, $x_j = 0$ otherwise; A_{ij} equals 1 if v_i is the source of e_j, -1 if v_i is the target of e_j, and 0 otherwise; b_i equals 1 if v_i is the start node, -1 if v_i is the goal, and 0 otherwise.

C. Inverse Shortest Path (ISP)

The goal of the inverse shortest path problem is to obtain a new weight vector $\boldsymbol{w}^{'}$ which turns a user-defined path $\boldsymbol{x}^{'}$ into the shortest path in the new graph—while making the least possible changes to the original weights. It can be formulated as a linear program [3]:

$$\min_{\boldsymbol{w}^{'}, \boldsymbol{\pi}, \boldsymbol{\lambda}} \quad \|\boldsymbol{w}^{'} - \boldsymbol{w}\|_1 \tag{2a}$$
$$\text{s.t.} \quad \sum\nolimits_i A_{ij}\pi_i = w_j^{'} \quad \forall_{j:x_j^{'}=1}, \tag{2b}$$
$$\sum\nolimits_i A_{ij}\pi_i + \lambda_j = w_j^{'} \quad \forall_{j:x_j^{'}=0}, \tag{2c}$$
$$\lambda_j \geqslant 0 \quad \forall_{j:x_j^{'}=0}, \tag{2d}$$
$$\boldsymbol{w}^{'} \in \mathbb{R}_{+}^{|E|}, \ \boldsymbol{\lambda} \in \mathbb{R}^{|E|}, \ \boldsymbol{\pi} \in \mathbb{R}^{|V|}. \tag{2e}$$

The first and second constraints are used to enforce complementary slackness conditions, and $\boldsymbol{\lambda}$ and $\boldsymbol{\pi}$ are the dual variables of the shortest path problem.

IV. Methods

A. Road network graphs

As in the ISP formulation (2), in this paper we model road networks as directed graphs. Furthermore, graphs are such that for every edge $e_j = (u, v) \in E$ there exists an edge in the opposite direction $(v, u) \in E$. Each edge e_j is associated with a variable $n_j \in \{0, 1\}$ which indicates whether driving in that direction is normally impermissible due to traffic rules (i.e. one-way roads will have one of the edges impermissible $n_j = 1$), as well as a variable $c_j \in \{0, 1\}$ which indicates temporary road closure. We further assume that the objective of our road navigation system is to minimize travel time. Therefore, we model edge weights as travel times—defined as a function of driving speed, distance, temporary road closure, and normal direction impermissibility. We compute the weight of an edge as:

$$w_j = s_j l_j + n_j M + c_j M, \tag{3}$$

where

- $s_j = \theta_j^{-1}$ is the inverse of the current flow speed θ_j (obtained from live traffic data), which for simplicity is truncated to legal limits $\theta_j \in [0, \theta_j^{\max}]$, where $\theta_j^{\max}$ is the speed limit of road j;
- $l_j \in \mathbb{R}_{0+}$ is the edge length in meters;
- $n_j \in \{0, 1\}$ indicates whether driving on edge j is normally impermissible (depending on whether the road is one-way or two-way);
- $c_j \in \{0, 1\}$ indicates whether traffic on edge j is temporarily closed;
- and $M \approx \infty$ is a large number (e.g. $M = 10^6$).

In this paper we consider generating explanations for two kinds of why questions:

1) "Why don't you take path $\boldsymbol{x}'$ instead?", where $\boldsymbol{x}'$ is a user-defined path. We will call this the "full-path explanation problem".
2) "Why don't you go through v_w instead?", where $v_w \in V$ is a user-defined waypoint node. We will call this the "waypoint explanation problem".

B. Full-Path Explanation Problem (Why not path B?)

We model the full-path explanation problem as an ISP: where the goal is to find the lowest number of changes to a road network that lead the desired path to become optimal. These changes serve as an explanation, e.g. "the path is not B because there is traffic in road X and Y"; or "the path is not B because road Z is one-way". Our explanations consider speed limits, the presence of traffic (road congestion), normally permitted directions of travel, and temporary road closure.

To allow for an ISP formulation, we define the lifting of speed limits $b_j^{m'}$, the removal of congestion $b_j^{s'}$, the normal impermissibility of travel in an edge's direction n_j', and temporary road closure c_j' as boolean variables associated with each edge. The explanation-ISP will work by finding changes to these variables that lead to the desired path becoming optimal. We express the explanatory weight w'_j of an edge e_j as a function of these variables, by:

$$w'_j = \begin{cases} (b_j^{m'} \overline{m} + (1 - b_j^{m'}) m_j) l_j + (n'_j + c'_j) M & \text{if } \frac{m_j}{s_j} \geq r \\ (b_j^{s'} m_j + (1 - b_j^{s'}) s_j) l_j + (n'_j + c'_j) M & \text{if } \frac{m_j}{s_j} < r \end{cases}, \tag{4}$$

where $m_j = \frac{1}{\theta_j^{\max}}$ is the inverse of the speed limit of road j, $\overline{m}$ is the maximum possible speed limit in any road (typically the speed limit of highways), and $r \in [0, 1]$ is a parameter which defines the speed ratio at which a road is considered congested.

In simple terms, the equation makes it so that:

- if an edge is congested (i.e. $\frac{m_j}{s_j} < r$) then we allow congestion to be part of the explanation (by setting $b_j^{s'} = 1$ and thus removing congestion, i.e. setting traffic speed to the road's current speed limit);
- if an edge is not congested (i.e. $\frac{m_j}{s_j} \geq r$) then we allow the road's speed limit to be part of the explanation (by setting $b_j^{m'} = 1$ and thus lifting the road's speed limit to the maximum possible $\overline{m}$);
- regardless of congestion, we allow normal impermissibility of travel or temporary road closure to be part of the explanation (by setting n'_j or c'_j).

By replacing w'_j in the ISP formulation (2) by the equation above (4), the ISP becomes a Mixed-Integer ISP to which we call "Road-Graph Inverse Shortest Path" (RGISP):

$$\underset{\boldsymbol{n}', \boldsymbol{c}', \boldsymbol{b^{s'}}, \boldsymbol{b^{m'}}, \boldsymbol{\pi}, \boldsymbol{\lambda}}{\text{minimize}} \quad \|\boldsymbol{n}' - \boldsymbol{n}\|_1 + \|\boldsymbol{c}' - \boldsymbol{c}\|_1 + \|\boldsymbol{b^{s'}}\|_1 + \|\boldsymbol{b^{m'}}\|_1 \tag{5a}$$

$$\text{subject to}$$

$$\sum_i A_{ij} \pi_i = s'_j l_j + n'_j M + c'_j M, \quad \forall_{j : x'_j = 1 \,\wedge\, m_j/s_j < r}, \tag{5b}$$

$$\sum_i A_{ij} \pi_i = m'_j l_j + n'_j M + c'_j M, \quad \forall_{j : x'_j = 1 \,\wedge\, m_j/s_j \geqslant r}, \tag{5c}$$

$$\sum_i A_{ij} \pi_i + \lambda_j = s'_j l_j + n'_j M + c'_j M, \quad \forall_{j : x'_j = 0}, \tag{5d}$$

$$\lambda_j \geqslant 0, \quad \forall_{j : x'_j = 0}, \tag{5e}$$

$$n'_j = n_j, \;\; c'_j = c_j, \quad \forall_{j : x^*_j = 1 \,\wedge\, x'_j = 0}, \tag{5f}$$

$$n'_j = 0, \;\; c'_j = 0, \quad \forall_{j : x'_j = 1}, \tag{5g}$$

$$\boldsymbol{n}' \in \{0,1\}^{|E|}, \;\; \boldsymbol{c}' \in \{0,1\}^{|E|}, \;\; \boldsymbol{b^{s'}} \in \{0,1\}^{|E|}, \;\; \boldsymbol{b^{m'}} \in \{0,1\}^{|E|}, \;\; \boldsymbol{\lambda} \in \mathbb{R}^{|E|}, \;\; \boldsymbol{\pi} \in \mathbb{R}^{|V|}, \tag{5h}$$

where:

- $s'_j = b_j^{s'} m_j + (1 - b_j^{s'}) s_j$ as in (4)
- $m'_j = b_j^{m'} \overline{m} + (1 - b_j^{m'}) m_j$ as in (4)
- $\boldsymbol{n}$ is the normal impermissibility of travel in the original graph, and $\boldsymbol{n}'$ is the variable for the new graph;
- $\boldsymbol{c}$ is road closure in the original graph, and $\boldsymbol{c}'$ is the variable for the new graph.

To obtain the actual explanation from the output of RGISP (i.e. from the values $\boldsymbol{n'}$, $\boldsymbol{c'}$, $\boldsymbol{b^{s'}}$, $\boldsymbol{b^{m'}}$), we simply compare the original and new values (c_j vs c'_j, n_j vs n'_j, 0 vs $b_j^{s'}$, 0 vs $b_j^{m'}$) to detect which edge properties have changed in the new graph. Then, we fill in an explanation text-template with the names of the roads and the associated explanation, e.g.: "the desired path is not optimal because road X is congested/low-speed-limit/one-way/temporarily-closed".

C. *Waypoint Explanation Problem (Why not waypoint W?)*

Next, we propose a method to explain why the shortest-path does not traverse a user-provided waypoint v_w. For this purpose, we need to find whether there exists a small set of changes to the graph that lead to an optimal path $\boldsymbol{x}^*$ that crosses v_w, i.e.$\exists_j : x^*_j = 1 \wedge v_w \in e_j$.

One naive solution is to compute the shortest path through v_w given G by concatenating the shortest paths from the start to v_w and v_w to goal—and then apply RGISP. However, such a strategy would be sub-optimal, since a shorter explanation may exist which makes changes on edges not lying over the concatenated path. The strategy is also incomplete, since it may be impossible to make the concatenated path optimal, but there may exist another path travelling through v_w which can be made optimal.

Instead, we propose an anytime asymptotically-optimal method: RGISP*. The method works by successively calling RGISP on diverse paths that traverse waypoint v_w, until exhausting all possible paths that traverse the waypoint. It is anytime because it can be stopped at any point in time and reveal the least-change explanation found so far (obtained through one of the paths traversing v_w).

Pseudo-code for the algorithm is shown in Algorithm 1. Basically, first the algorithm will use the naive solution described in the previous paragraph to obtain an initial (sub-optimal) explanation (lines 4-7). Then, while the runtime of the algorithm is below a user-provided limit $t_{\max}$, it will obtain a new path through the waypoint (line 9 "*NextPath*") and run RGISP on this path (line 11). The algorithm keeps track of the lowest-length explanation found so far (lines 12-13) in order to return it once the runtime has expired. Function ''*ShortestPathThroughWaypoint*" simply concatenates the shortest path from the start to the waypoint, with the shortest path from the waypoint to the goal.

This anytime algorithm relies on a function "*NextPath*" which incrementally enumerates all paths through waypoint v_w. For this purpose, we use an adapted version of the diverse path finding method of [4], as shown in Algorithm 2. Essentially, the algorithm iteratively and randomly removes edges that lie along the shortest-paths-through-the-waypoint. To do this, it searches over graphs using a branching factor b, where each child graph will remove an edge from its parent graph (lines 8-10). Edges are picked by random sampling from the shortest-path-through-the-waypoint of the parent graph—in order to invalidate that path and lead to a new shortest-path-through-the-waypoint. To guarantee completeness (i.e. that all paths through the waypoint are eventually obtained), we restart the procedure once the graph queue is exhausted (lines 3-5). The search begins from a graph G^o, which is obtained from G by making traffic along all edges permissible $c_j = 0 \, \forall_j$ (line 3 "*OpenAllEdges*"). This is so

Algorithm 1 RGISP*

Input: Road graph $G = (V, E)$; start, waypoint and goal nodes, $v_s, v_w, v_g \in V$; maximum runtime $t_{\text{max}} \geqslant 0$.
Output: Explanation graph G^*; explanation length z^*.

$U \leftarrow \textit{EmptyQueue}()$
$G^* \leftarrow \emptyset; \quad z^* \leftarrow \infty$
$\boldsymbol{x}_p \leftarrow \textit{ShortestPathThroughWaypoint}(G, v_s, v_w, v_g)$
if $\boldsymbol{x}_p$ **not empty then**
  $(G^e, z) \leftarrow \textit{RGISP}(G, \boldsymbol{x}_p)$
  if $z \neq$ **null** $\wedge\, z < z^*$ **then**
    $G^* \leftarrow G^e; \quad z^* \leftarrow z$
while $\textit{Runtime}() \leqslant t_{\text{max}} \wedge z^* > 0$ **do**
  $\boldsymbol{x}'_p, U \leftarrow \textit{NextPath}(G, v_s, v_w, v_g, U)$
  if $\boldsymbol{x}'_p$ **not empty then**
    $(G^e, z) \leftarrow \textit{RGISP}(G, \boldsymbol{x}'_p)$
    if $z \neq$ **null** $\wedge\, z < z^*$ **then**
      $G^* \leftarrow G^e; \quad z^* \leftarrow z$
return G^*, z^*

as to be able to obtain explanations that focus on traffic-permissibility (i.e. which change permissibility in one or more edges, thus obtaining a new shortest path that traverses those edges and satisfies the user's waypoint expectation).

D. Map pre-processing

Map services such as OpenStreetMap[1] typically provide road network data in multi-directed graph form, to account for locations (e.g. squares) where there are multiple ways to traverse between adjacent nodes (e.g. around the square clockwise or counter-clockwise). Without loss of generality, we convert multi-directed graphs to directed graphs as a pre-processing step. We do this by creating intermediate nodes along multi-directed edges.

V. Results

A. Experimental Setup

We conducted several experiments to demonstrate and evaluate our explanation methods. The experiments use a road graph of 1km radius, centered around King's College London Bush House campus, London, UK, and obtained at 1:30 am on the 7th of July, 2022. We used OSMnx[2] to obtain road graph data from OpenStreetMap, and we used TomTom API[3] to gather real-time data (i.e. road closure and current traffic speed). All our experiments use a traffic ratio parameter $r = 3/4$, and they use the United Kingdom's maximum legal speed limit of $\overline{m} = 70$ miles per hour for speed-limit explanations. For the anytime algorithm RGISP* we set a branching factor of $b = 2$ and a computation time budget of $t_{\text{max}} = 5$ minutes. We provide source code and a working demo at `https://github.com/khalid-alsheeb/explainable-road-navigation`.

[1]https://www.openstreetmap.org/
[2]https://osmnx.readthedocs.io
[3]https://developer.tomtom.com

Algorithm 2 NextPath (Incremental Path Enumeration)

Input: Road graph $G = (V, E)$; start, waypoint and goal nodes, $v_s, v_w, v_g \in V$; auxiliary queue U.
Parameters: Branching factor $b \geqslant 1$.
Output: Path $\boldsymbol{x}_p^{\text{next}}$ which traverses start, waypoint and goal nodes; updated queue U.

$\boldsymbol{x}_p^{\text{next}} \leftarrow \emptyset$
if U **empty then**
  $G^o \leftarrow \textit{OpenAllEdges}(G)$
  $\boldsymbol{x}_p^o \leftarrow \textit{ShortestPathThroughWaypoint}(G^o, v_s, v_w, v_g)$
  $\textit{Enqueue}(U, (\boldsymbol{x}_p^o, G^o))$
$(\boldsymbol{x}_p^u, G^u) \leftarrow \textit{Dequeue}(U)$
for i **in** $1, \dots, b$ **do**
  $e \leftarrow \textit{UniformSampling}(\boldsymbol{x}_p^u)$
  $E' \leftarrow \textit{AddObstacles}(E^u, e)$
  $G' \leftarrow (V, E')$
  $\boldsymbol{x}'_p \leftarrow \textit{ShortestPathThroughWaypoint}(G', v_s, v_w, v_g)$
  if $\boldsymbol{x}'_p$ **not empty then**
    $\textit{Enqueue}(U, (\boldsymbol{x}'_p, G'))$
    if $\boldsymbol{x}_p^{\text{next}}$ **not empty then**
      $\boldsymbol{x}_p^{\text{next}} \leftarrow \boldsymbol{x}'_p$
return $\boldsymbol{x}_p^{\text{next}}, U$

We used CVXPY [16] to formulate the optimization problems, and Gurobi[4] as the solver. All experiments were run on a 2018 MacBook Pro, 2.6 GHz 6-Core Intel Core i7 processor, 16GB RAM, macOS Big Sur OS.

B. Example of Full-Path Explanation

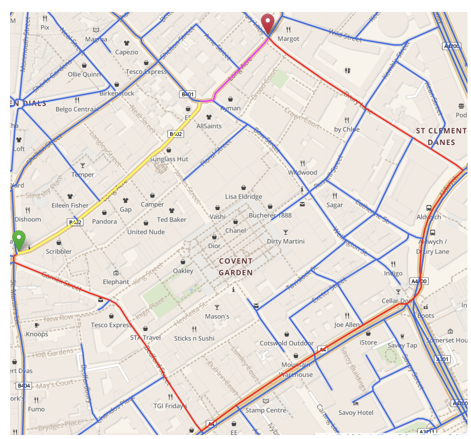
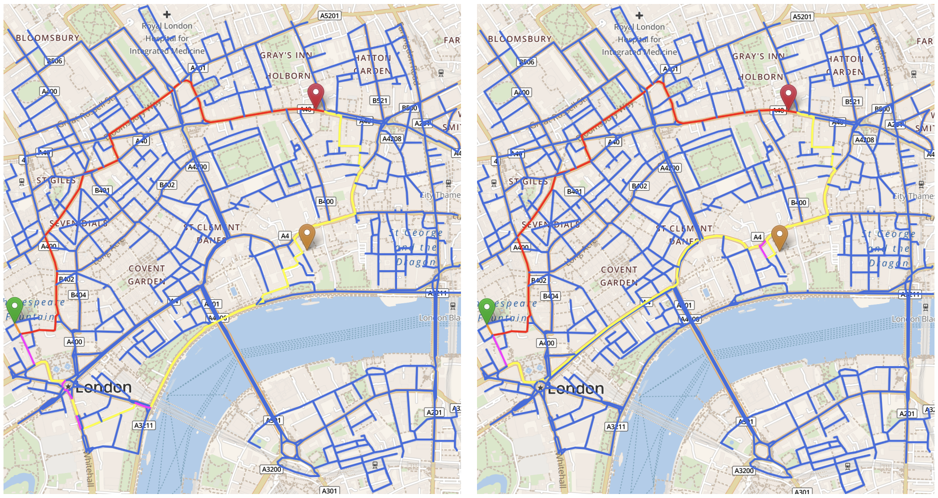
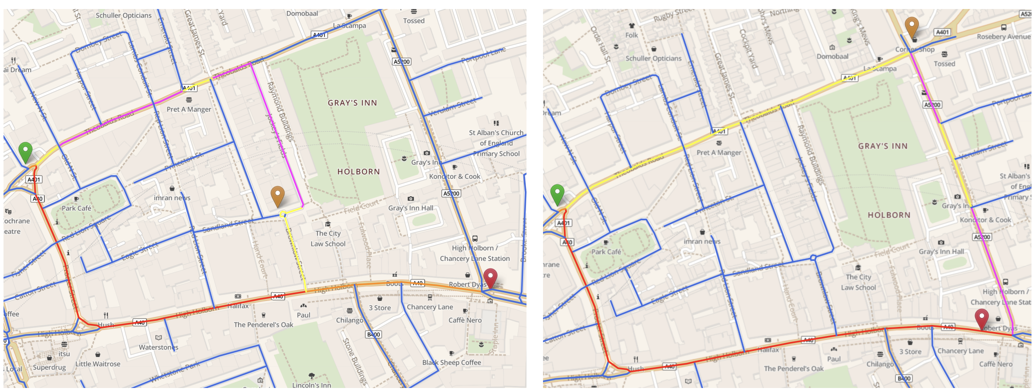


Fig. 1: Example of a full-path explanation problem. A user asks for a path from the green marker to the red marker. Graph edges are shown in blue, and the optimal path is shown in red. The user is already partially familiar with the map, and is confused by the path proposed. The user then manually provides the path in yellow and asks "why not this path?". Our method explains that is because the roads in pink are either closed or one-way roads.



(a) Naive solution by RGISP (b) Solution by RGISP*

Fig. 2: Example waypoint explanation. A user asks for a path from the green marker to the red marker. Graph edges are shown in blue, and the optimal path is shown in red. The user then manually provides a waypoint (yellow marker) and asks "why not a path through the yellow marker?". Our method explains that is because the roads in pink are one-way roads. A naive solution using RGISP (with the shortest path through the waypoint as desired path) leads to a complex explanation with many road changes, while our anytime asymptotically-optimal RGISP* leads to a simpler explanation (length 11 vs 4).



(a) Heavy traffic and a one-way road. (b) Speed limit

Fig. 3: Example waypoint explanations using RGISP*. A user asks for a path from the green marker to the red marker. Graph edges in blue, optimal path in red. The user then manually provides a waypoint (yellow marker) and asks "why not a path through the yellow marker?". Our method explains that is (a) because Theobalds Road (pink) has heavy traffic and Jockey's Fields (pink) is a one-way road; (b) because Gray's Inn Road (pink) has a low speed limit. Had these facts not been true then the shortest paths would have been the ones in yellow—satisfying the user's waypoint preferences. Orange sections are those that are common between the original and waypoint-satisfying paths.

Figure 1 shows an example of a full-path explanation obtained with RGISP. For this purpose, we manually provided as input the start and goal locations of a hypothetical trip (green and red marker respectively). The shortest path between these two nodes is shown in red. This path is very long compared to a straight-line path, and so we manually provided a shorter straight-line path (shown in yellow), and ran RGISP. RGISP provides an explanation for the question "why not this path (in yellow) instead?". As described previously, our method does this by finding the smallest set of changes to the graph that lead the yellow path to become optimal. RGISP provided the following explanation:

> "The desired path is not optimal because Endell Street is currently closed; and Long Acre (0-1) is a one-way road."

Endell Street and Long Acre are shown in pink on Fig. 1.

C. *Examples of Waypoint Explanation*

Fig. 2 shows an example of a waypoint explanation obtained with our methods. We manually provided as input the start and goal locations of a hypothetical trip (green and red marker respectively). The shortest path between these two nodes is shown in red. We then manually provided an expected waypoint (yellow marker), so as to ask the

[4]https://www.gurobi.com/products/gurobi-optimizer/

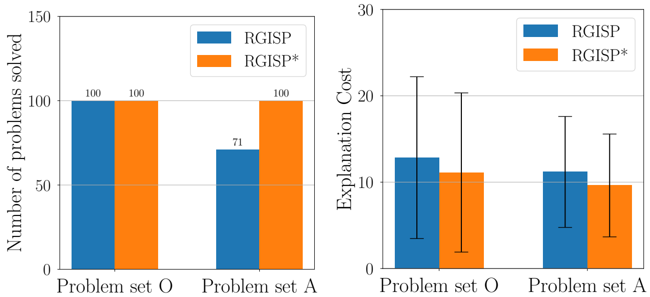


Fig. 4: Success rate and explanation cost, on waypoint explanation problems, of the anytime asymptotically-optimal RGISP* vs a naive RGISP strategy.

question "why not a path through the yellow marker?". A naive solution using RGISP, with the shortest-path-through-the-waypoint as desired path, leads to an explanation with many road changes (11 changes shown in pink on Fig. 2(a)):

> "The desired path is not optimal because Whitcomb Street (0-1); Charing Cross (0-3); Whitehall (0-2); and Northumberland Avenue (0-1) are one-way roads."

Numbers in brackets indicate sections of the roads. The new shortest path is shown in yellow in the figure.

Fig. 2(b) shows the explanation computed by our anytime asymptotically-optimal RGISP*, after a 20s computation time budget. The anytime algorithm managed to obtain a considerably shorter (i.e. lower-cost) explanation with only 4 changes:

> "The desired path is not optimal because Strand is not closed to traffic; and Whitcomb Street (0-1) and Milford Lane are one-way roads."

Finally, in Fig. 3, we show examples of user queries that would lead to explanations centered on traffic (Fig. 3(a)):

> "The desired path is not optimal because Theobalds Road (0-3) has heavy traffic; and Jockey's Fields is a one-way road."

and on a low speed limit (Fig. 3(b)):

> "The desired path is not optimal because Gray's Inn Road (0-1) has a speed limit of 20 mph."

D. *Quantitative Evaluation on Waypoint Problems*

We quantitatively evaluated our methods using 200 randomly generated waypoint explanation problems (Problem set O and A). Problem set O (waypoint problems solvable by the naive RGISP strategy) consists of 100 explanation problems obtained by randomly selecting start, goal and waypoint nodes, solving the corresponding problem by RGISP (i.e. with the shortest-path-through-the-waypoint as desired path), and storing the problem as part of O if RGISP was able to obtain a solution. Problem set A (waypoint problems solvable by the anytime algorithm RGISP*) consists of 100 explanation problems obtained by randomly selecting start, goal and waypoint nodes, solving the corresponding problem with RGISP*, and storing the problem as part of A if RGISP* was able to obtain a solution within its time budget.

Fig. 4 shows that the anytime algorithm was able to solve all 100 waypoint explanation problems that the RGISP strategy was able to solve (set O). However, the naive RGISP strategy was only able to solve 71 problems out of the 100 that RGISP* could solve. The right side of Fig. 4 further shows that the average explanation cost (i.e. value of the optimization objective in equation (5)) of both strategies. The figure shows that the anytime method, as expected by design, leads to lower-cost explanations.

Fig. 5 shows some examples of waypoint explanation problems that can be solved by both strategies. The first graph shows how, in some situations, one extra iteration by the anytime algorithm is enough to provide a better explanation. The second and third graphs show problems where more iterations are required to refine the explanation.

VI. Conclusion

In this paper we proposed two algorithms, based on inverse optimization methods, to solve two explanation problems in road navigation systems: 1) the full-path explanation problem, where the goal is to answer why a user-specified path is not optimal; and 2) the waypoint explanation problem, where the goal is to answer why the optimal path does not travel through a user-specified waypoint. In order to automatically generate such explanations, we rely on an inverse shortest path method that finds minimal-changes to road network parameters (i.e. congestion, permitted travel direction, temporary closure, speed limits) that lead an input path to become optimal. For full-path explanation problems this input path is the user-specified path, while for waypoint problems it is the set of all paths that travel through the user-specified waypoint. To obtain this set of paths incrementally in an anytime manner, we adapted an existing diverse shortest paths algorithm. We then demonstrated how the algorithms are capable of computing various kinds of explanations through concrete examples on OpenStreetMap data. We also showed how the anytime asymptotically-optimal algorithm for waypoint problems (RGISP*) leads to shorter explanations than naive strategies, and their length decreases with computation time as designed.

This paper thus paves the way for explainable road navigation systems: such as route planners for AVs and traditional vehicles, as well as road navigation apps for smartphones.

One important direction of research includes the design of computationally faster explanation algorithms, for example by the use of incremental optimization methods such as those proposed in [2]. In order to guarantee good user experience, future research should involve users of navigation systems in the design of the "why questions" and algorithmic explanations. This could be done, for example, through user studies that identify the kind of mental models held by road users, the level of detail that is most appropriate in explanations, and the navigation parameters most relevant for explanation. Concretely, we should look to identify the

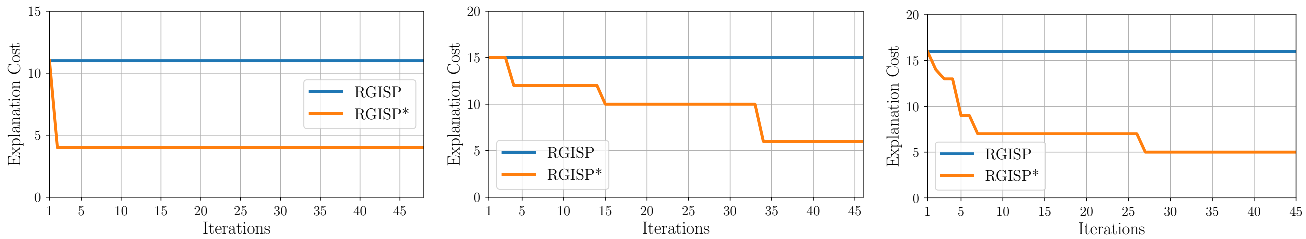


Fig. 5: Cost of waypoint explanations obtained by a naive RGISP strategy, and by the anytime asymptotically-optimal RGISP*. The cost indicates the number of changes made to graph variables, as seen in equation (5).

relative usefulness of each parameter we have used (speed limits, road closure, permitted travel directions), but also to identify the need of other parameters such as number of turns, traffic lights, average traffic speed, congestion probability, etc. User studies should also be conducted to understand whether the explanations introduced in this paper improve users' mental models of maps (similarly to work in robotics or computer game settings [2]), and whether this transfers to better navigation decisions without the system—or to greater delegation to, and eventually dependency on, the navigation systems.

REFERENCES

[1] Google maps metrics and infographics. [Online]. Available: https://sites.google.com/a/pressatgoogle.com/google-maps-for-iphone/google-maps-metrics

[2] M. Brandao, A. Coles, and D. Magazzeni, "Explaining path plan optimality: Fast explanation methods for navigation meshes using full and incremental inverse optimization," in *Proceedings of the International Conference on Automated Planning and Scheduling (ICAPS)*, Aug 2021, pp. 56–64.

[3] R. K. Ahuja and J. B. Orlin, "Inverse optimization," *Operations Research*, vol. 49, no. 5, pp. 771–783, 2001.

[4] C. Voss, M. Moll, and L. E. Kavraki, "A heuristic approach to finding diverse short paths," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2015, pp. 4173–4179.

[5] L. Chazette, V. Klös, F. Herzog, and K. Schneider, "Requirements on explanations: a quality framework for explainability," in *2022 IEEE 30th International Requirements Engineering Conference (RE)*. IEEE, 2022, pp. 140–152.

[6] D. Omeiza, H. Webb, M. Jirotka, and L. Kunze, "Explanations in autonomous driving: A survey," *IEEE Transactions on Intelligent Transportation Systems*, 2021.

[7] T. Schneider, J. Hois, A. Rosenstein, S. Ghellal, D. Theofanou-Fülbier, and A. R. Gerlicher, "Explain yourself! transparency for positive ux in autonomous driving," in *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*, 2021, pp. 1–12.

[8] T. Miller, "Explanation in artificial intelligence: Insights from the social sciences," *Artificial Intelligence*, vol. 267, 2019.

[9] H. Huang, T. Mathis, and R. Weibel, "Choose your own route–supporting pedestrian navigation without restricting the user to a predefined route," *Cartography and Geographic Information Science*, vol. 49, no. 2, pp. 95–114, 2022.

[10] I. Fellner, H. Huang, and G. Gartner, "turn left after the wc, and use the lift to go to the 2nd floorgeneration of landmark-based route instructions for indoor navigation," *ISPRS International Journal of Geo-Information*, vol. 6, no. 6, p. 183, 2017.

[11] M. Brandao and Y. Setiawan, "'why not this mapf plan instead?' contrastive map-based explanations for optimal mapf," in *ICAPS 2022 Workshop on Explainable AI Planning (XAIP)*, June 2022.

[12] M. Fox, D. Long, and D. Magazzeni, "Explainable planning," *arXiv preprint arXiv:1709.10256*, 2017.

[13] T. Chakraborti, S. Sreedharan, Y. Zhang, and S. Kambhampati, "Plan explanations as model reconciliation: Moving beyond explanation as soliloquy," in *International Joint Conference on Artificial Intelligence (IJCAI)*, 2017, pp. 156–163.

[14] B. A. Miller, Z. Shafi, W. Ruml, Y. Vorobeychik, T. Eliassi-Rad, and S. Alfeld, "Pathattack: Attacking shortest paths in complex networks," in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 2021, pp. 532–547.

[15] H. Aljazzar and S. Leue, "K*: A heuristic search algorithm for finding the k shortest paths," *Artificial Intelligence*, vol. 175, no. 18, pp. 2129–2154, 2011.

[16] S. Diamond and S. Boyd, "CVXPY: A Python-embedded modeling language for convex optimization," *Journal of Machine Learning Research*, vol. 17, no. 83, pp. 1–5, 2016.