Qualitative Place Maps for Landmark-based Localization and Navigation in GPS-denied Environments

Hua Hua*
hua.hua@anu.edu.au
Research School of Computer Science
Australian National University

Peng Zhang*
p.zhang@anu.edu.au
Research School of Computer Science
Australian National University

Jochen Renz
jochen.renz@anu.edu.au
Research School of Computer Science
Australian National University

ABSTRACT

GPS-based services (e.g. Google Maps) are very popular in our daily life, while there are still many GPS-denied environments (e.g. indoor and underground scenarios) in which they cannot be used. In these situations, localization and navigation are still important, for example in emergency evacuation or indoor navigation. In this paper we aim to solve the problem of localizing and navigating humans or robots in GPS-denied environments based on landmarks. Our work is inspired by human daily communications about localization and navigation, for example someone who has been to a shopping mall many times can localize and guide another person to get to a certain shop via conversations over the cellphone. Our goal is to build a system with the same capability. We propose a system that relies on qualitative information of places (e.g. the direction relations between landmarks involved in route descriptions), where localization can be achieved in an interactive manner and by analyzing observations provided by users. Our system decides the "best" route from one place to another by three factors: the number of landmarks; the number of ambiguous turns; and the qualitative distance (e.g. near and far). According to the experimental results, the number of requeries in the interactive localization process is acceptable and our route planning algorithm outperforms previous methods in several cases.

CCS CONCEPTS

• **Information systems** → **Location based services**; **Geographic information systems**; • **Computing methodologies** → *Spatial and physical reasoning*.

KEYWORDS

Interactive localization, reliable navigation, landmark-based navigation, qualitative relative directions, qualitative spatial reasoning

ACM Reference Format:
Hua Hua, Peng Zhang, and Jochen Renz. 2019. Qualitative Place Maps for Landmark-based Localization and Navigation in GPS-denied Environments. In *27th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems (SIGSPATIAL '19), November 5–8, 2019, Chicago, IL, USA.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3347146.3359107

*Both authors contributed equally to this research.

SIGSPATIAL '19, November 5–8, 2019, Chicago, IL, USA

ACM ISBN 978-1-4503-6909-1/19/11…$15.00
https://doi.org/10.1145/3347146.3359107

1 INTRODUCTION

How to correctly move from the current place to the destination is fundamental in our daily life. This can be called the navigation problem, which is also a research problem that has been widely and extensively discussed in several different fields (e.g. Geographical Information System, Computer Science and Robotics). The following sub-problems have to be considered in most cases though researchers from different fields have different focuses:

- **Map.** Are online map services (e.g. Google Maps) available in the current environment? It is ideal to have a complete and precise map at-hand but navigation is still possible without such maps. For example, in a scenario where an out-of-towner asks a local resident how to get to a certain place, the local resident can describe a route by referring to the imaginary map in his/her mind.
- **Localization.** If GPS is available, it is trivial to locate the current position. However, there are still many GPS-denied environments where other forms of localization mechanisms are still required: indoor and underground environments where there is only unstable or no GPS-signal (e.g. shopping malls and mines, or inner cities with many high-rises); battlefield situations where GPS is jammed, or events where it is not allowed or not possible to use GPS (e.g. orienteering).
- **Route Planning.** How to find the best route from the origin to the destination? Another problem that might be equally important is how to define the "best" route? Dijkstra [7] is the most popular algorithm to find the shortest path. However, in some cases the "simplest" route is more preferable [8]. "Simplest" can be defined in different ways. For example, the route with fewest turns or the route that is the easiest to be described [28].
- **Route Instruction.** Route instructions can be either textual (e.g. route descriptions in natural language) or pictorial (e.g. routes that are illustrated in a sketched map or on the screen of a mobile device). How to describe a route in a less ambiguous way? For example, at an intersection there is more than one right turn, it is better to indicate which right turn to take (e.g. "next intersection take the first right turn" rather than just "next intersection turn right").
- **Decision points and Landmarks.** Agents have to make decisions (e.g. go straight or make turns) at decision points (e.g. intersections or corners). Landmarks are salient places so that their positions can be regarded as "known" and be used to locate other places. Decision points are indispensable in

navigation and landmarks are helpful in simplifying route instructions and eliminating ambiguities [23].

Agent. Humans, robots or vehicles that follow and execute certain route instructions to get from one place to another will be called "agents" [34] in this paper. Agents can be different in several aspects. Except for moving ability and the form of route instructions that can be utilized, agents can be also different in cognitive ability (e.g. whether they are able to recognize landmarks).

Re-query. It is possible that an agent is lost during a navigation process. In such cases re-queries are necessary so that agents can be re-located and the navigation process can continue by following a re-planned route.

In this paper, we will focus on the problem of localizing and navigating agents in GPS-denied environments based on landmarks. We also assume there is neither a precise and complete map nor an online map service like Google Maps available to the user for the area we are navigating in [35, 38]. Solving this problem is important for indoor navigation such as in subway stations or shopping centres, but also in extreme scenarios like natural disasters where quickly and reliably navigating to safe places can save lives [37]. For example, when there is a fire in a shopping mall, how to quickly find the route to the Exit from the current place (e.g. the Converse shop) only given the following two previously recorded route descriptions?

(1) Can you see the Adidas shop? Go there and turn left, you can see the Burberry shop. At the corner near Burberry turn left and walk for about 1-2 minutes and you will arrive at the Converse shop.

(2) Continue along this way until you can find the Burberry shop. At the elevator next to it turn right and go straight for about 50-60 meters where you can see Domino's Pizza. The Exit is very close to Domino's. Just make a slight right turn.

Such descriptions can be found in scenarios where people are asked to describe routes from memory. In the following we will describe a navigation system that can locate and navigate in GPS-denied environments given just incomplete and imprecise knowledge about the positions of places like the examples given above. Based on the above two route descriptions, our system should be able to infer a route from the Converse shop to the Exit.

The rest of this paper is organized as follows: in Section 2 we introduce related work; in Section 3 we describe how to construct a map of places that is helpful in the process of localization (Section 4) and navigation (Section 5); in Section 6 we demonstrate how our navigation system can be used by solving the problem of how to get to the Exit from the Converse shop; our solution is evaluated by experiments in Section 7; In Section 8 we conclude the paper and also discuss future work.

2 RELATED WORK

Dijkstra's algorithm [7] (or the shortest path algorithm) is to find the shortest path between two nodes in a directed graph where edges are labelled with weights (by default the distance between the two nodes but also can be other metrics like travel time). Following this pattern, directed graphs are also used to represent real-world traffic networks (where there will be an edge from one node to another if there is a realistic path in the traffic network) [8, 14] because it is convenient for further analysis.

In [14], the shortest most reliable path algorithm (which is an adapted version of the shortest path algorithm) is provided to find the most reliable path. Shorter path is more preferable to break ties when two paths are equally reliable. The unreliability of a path is calculated by summing the unreliability of all turns involved in this path and the unreliability of a turn is decided by the number of other turns that are instruction equivalent at the same intersection. For example, when arriving at a certain intersection from a certain direction, there are two turns on the left. The unreliability of each left turn is 1 because for each of them there is one other instruction equivalent turn.

In contrast, the simplest path algorithm [8] aims to find the simplest path under the assumption that it is simpler to describe paths with less number of turns or turns in less complex intersections. The complexity of different intersections has been discussed in [36]. The simplest path algorithm is also adapted from the shortest path algorithm. The computational complexity of both algorithms is $O(|E|^2)$ or $O(|V|^2)$ where $|E|$ is the number of edges and $|V|$ is the number of nodes in the directed graph under the assumption that the directed graphs that representing traffic networks are sparse (then it is safe to have $O(|E|) = O(|V|)$).

Following the simplest path algorithm, Richter and Duckham [28] propose the simplest instruction algorithm and claim that by using their algorithm the best paths can be described with much smaller number of instructions (on average a 50% decrease) when compared to both of the shortest path algorithm and the simplest path algorithm with an acceptable increase of the path length (on average a 10% increase). This improvement relies on considering the complexity of different intersections, referring to landmarks and applying the technique of spatial chunking. Landmarks are frequently referred to in route instructions because they can be used to indicate crucial turns (e.g. "turn right at the bus station"); locate other places (e.g. "the library is to the left of the bus station"); and ensure the route instructions are being correctly followed (e.g. "if you can find a bus station then you are in the right way") [28]. Spatial chunking is used to describe several continuous decision points in one instruction rather than multiple instructions. For example, "turn left at the second traffic light" rather than "go straight at the first traffic light and then turn left at the second".

There are quite a few discussions on landmarks since they are frequently utilized in navigation as mentioned above. As pointed out by [23], landmarks are helpful in unfamiliar environments and people tend to use landmarks more frequently when they have to make turns. A review of research on navigation with landmarks can be found in [9] where three approaches for landmark identification have been introduced and also a new navigation model based on landmarks has been proposed. Their landmark navigation model actually focuses on generating simple route instructions that include landmarks. In the navigation system presented in [30], the complete and precise map of a building is given by an indoor geographic information system. Visual landmarks (i.e. landmarks that can be recognized using vision) are used for localization.

There is also research on generating more sophisticated "human-like" landmark-based route instructions. The developed prototype "IndoorNav" [29] can locate a user by requiring them to scan the nearest QRCode and provide route instructions to the destination by considering all the direction relations of visible landmarks along a certain route. Grouping (similar to spatial chunking) is used to simplify textual instructions. The Indoor Landmark Navigation Model (ILNM) proposed in [11] relies on spatial objects that are in common categories (e.g. doors or elevators) rather than landmarks with unique or salient visual, semantic or structural characteristics. Deep learning methods can also be helpful in generating route instrcutions in natural languages as described in [6, 24].

According to [5], the majority of landmark descriptions are in the form of direction relations. Duckham et al. [9] also argue that numerical descriptions of distance or turning angles are very rare in human route descriptions. To be closer to human understanding of spatial knowledge used in navigation (e.g. the direction relations and topological relations between landmarks) and to achieve better human-robot interaction, it is natural to describe landmarks with their qualitative spatial relations. Qualitative Spatial Representation and Reasoning is a sub-field of Knowledge Representation and Reasoning in Artificial Intelligence that focuses on the representation and reasoning of qualitative spatial relations between spatial entities [4, 26]. Qualitative direction calculi (which are systematic models of sets of qualitative direction relations) are helpful in modeling directions that are heavily used in route descriptions. For example, the double-cross calculus [13] that can model the direction relation between three points is applied in [18] to generate route descriptions to support the communication between a driver and an intelligent wheelchair; in [10, 22] $\mathcal{OPRA}_m$ (a qualitative calculus that can deal with the relative direction relations between pairs of oriented points) has been used for navigating agents.

Such qualitative direction relations can be converted from quantitative data in geographical information systems (e.g. OpenStreetMap data) as in [22]; or be extracted from real-world natural language place descriptions (e.g. the two descriptions in the introduction) in the form of "Place A Relation Place B" [21] and be formalized as place graphs [2, 33].

More and more robotics researchers have made efforts to handle qualitative input (e.g. natural language direction description [18, 19] or sketched maps [1, 3, 31]) because qualitative representations are more suitable for human-robot interaction though traditionally most navigation strategies in this field are based on quantitative input (e.g. metric data collected from sensors or cameras, OpenStreetMap data, or floor plans). Their research focuses more on interpreting qualitative input and applying it for path planning rather than route planning.

Note that in some cases the two concepts of "path" and "route" are used as synonyms (e.g. the shortest path and the shortest route). In this paper, to avoid ambiguity, "path" and "route" are regarded as different terms in the sense that route planning is to find the "best" route from the origin to the destination though the definition of "best" varies and in contrast path planning is to deal with the problems about how to precisely move (e.g. motion trajectory planning; and obstacle avoidance).

In this paper we focus on qualitative navigation (i.e. solving the navigation problem by qualitative input [12, 20]), which is different from quantitative navigation that heavily relies on quantitative input. To the best of our knowledge, our solution is the first that combines the following features: qualitative knowledge (e.g. the direction and distance relations between places) as input; landmark-based; interactive localization; and deciding the best route by considering number of landmarks, number of ambiguous turns and qualitative distance. We believe our solution is different from previous solutions in reliability and is closer to how humans are doing it and therefore is more intuitive for humans to use.

3 QUALITATIVE PLACE MAP

In a GPS-denied environment, localization and navigation are still possible by referring to the descriptions of places from people who are familiar with this area. Note that in this paper places can be either landmarks (e.g. the Adidas shop) or non-salient places (e.g. turns or elevators, or not so well-known branded shops). A qualitative place map is a map of places in a qualitative form. In this section, we will introduce its formal definition and also provide construction algorithms in different scenarios.

Definition 3.1 (Qualitative Place Map). A qualitative place map (QPM) is a directed graph whose vertices are places and the edges between vertices are labelled with the qualitative knowledge about places. Namely a QPM is a tuple (V, E) where $V = \{v_1, v_2, ..., v_n\}$ is the set of vertices (n is the number of vertices) and $E = \{e_1, e_2, ..., e_m\}$ is the set of edges (m is the number of edges).

In a QPM, each vertex $v \in V$ has two attributes: the name of its corresponding place and whether the place is a landmark. The naming function is defined to be $pn : V \longrightarrow S$ where S is a set of place names; and the landmark function is $lm : V \longrightarrow \{t, f\}$ (t: true and f: false). For example let v_1 represent the "Adidas shop" and v_2 represent the "corner" in the first description then $pn(v_1) = Adidas$, $pn(v_2) = corner$, $lm(v_1) = t$ and $lm(v_2) = f$. An intuitive landmark judging strategy that only relies on place names is used in this paper while more sophisticated techniques can be found in the relevant research introduced in Section 2: in a certain environment places with unique and semantic names are landmarks (e.g. the Adidas shop or the Big Clock) while places with names that are not unique, not well-known, or only indicate their functions are not (e.g. turns, elevators or doors).

Each edge $e \in E$ is associated with a label consisting of six elements. Namely, $e = (v_s, v_e, r_{rea}, r_{vis}, R_{dir}, r_{dis})$ where $v_s, v_e \in V$ indicate the starting vertex and the ending vertex respectively, $r_{rea} \in \{t, f\}$ (t: true and f: false) denotes whether v_e is directly reachable from v_s (namely there is a direct path between them), $r_{vis} \in \{t, f\}$ denotes whether v_e is visible from v_s, R_{dir} is the set of qualitative direction relations between v_s and v_e and r_{dis} is the qualitative distance relation between them.

3.1 Direction Relations

The qualitative direction relation calculus used in this paper Relative Direction Relation ($\mathcal{RDR}_m$) [16] is an adaptation of the $\mathcal{STAR}_m$ calculus [25] and the double-cross calculus [13] to model the direction relations of place triples with arbitrary granularities. As the example in Figure 1a where there are three places A, B and C and $m = 4$, the 2D plane is divided into $4 * 4$ sectors by 4 lines and these sectors are indexed from 0 to $4 * 4 - 1$ (i.e. 15). Sectors indexed with even numbers are even sectors (half lines in the 2D plane) and those indexed with odd numbers are odd sectors (convex regions in the 2D plane). The direction relation between A, B and C is 1 because C

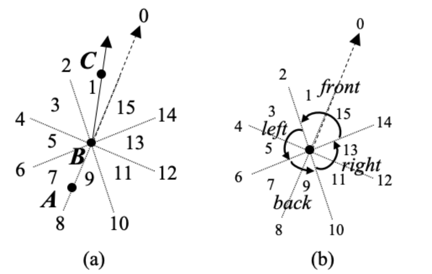

Figure 1: (a) By using the default qualitative direction relation calculus $\mathcal{RDR}_4$ the direction relation between A, B and C is 1. (b) The default direction relation labelling function that assigns the 16 relations with the labels "front, left, back and right".

is in Sector 1 given B as the centroid and the orientation of Sector 0 the same as $A \rightarrow B$. The Direction Relation Labelling function is defined to be: $l_{dir} : Z_{4m} \longrightarrow L_{dir}$ where $Z_{4m} = \{0, 1, ..., 4m - 1\}$ and L_{dir} is a set of direction relation labels in natural language (e.g. $\{front, left, back, right\}$). Its converse function is defined as $l_{dir}^{-1} : L_{dir} \longrightarrow Z_{4m}$. In this paper the default qualitative direction relation calculus is $\mathcal{RDR}_4$ and the default l_{dir} function (as in Figure 1b) is

- $l_{dir}(0, 1, 15) = front$
- $l_{dir}(2, 3, 4, 5, 6) = left$
- $l_{dir}(7, 8, 9) = back$
- $l_{dir}(10, 11, 12, 13, 14) = right$

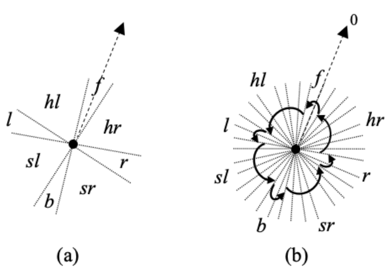

Figure 2: (a) The sector model used by Klippel et al. f: front, hl: half left, l: left, sl: sharp left, b: back, sr: sharp right; r: right; hr: half right. (b) An equivalent sector model with m = 16 and a dedicated direction relation labelling function.

The value of m and the corresponding l_{dir} function are configurable because in different scenarios the size of sectors and the natural language terms to describe each sector might vary. For example, the sector model in [17] is shown in Figure 2a: the 2D plane is not equally divided into even and odd sectors and "half left", "left" and "sharp left" are used instead of just "left". Their model can be represented by $\mathcal{RDR}_m$ with a greater m and adjusting the l_{dir} function as in Figure 2b (which originates from the mapping strategy in [22]).

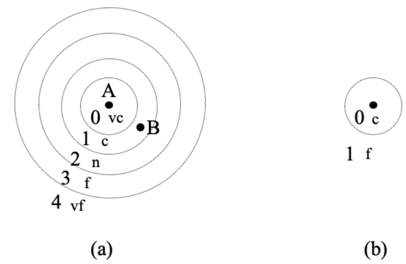

Figure 3: (a) The default qualitative distance calculus $\mathcal{QDR}_5$. vc: very close, c: close, n: neural, f: far, vf: very far. (b) Another qualitative distance calculus $\mathcal{QDR}_2$. c: close, f: far.

3.2 Distance Relations

Similarly, the qualitative distance calculus used in this paper Qualitative Distance Relation ($\mathcal{QDR}_m$) is first discussed in [15] to model the qualitative distance relations between places with arbitrary granularity. As in Figure 3a the 2D plane is divided into a circle and several other circular rings based on how far the points are from the center. These regions are indexed from 0 to 5 − 1 (i.e. 4). The distance relation between A and B is 1 because B is in Region 1 of A (Figure 3a). The Distance Relation Labelling function is defined to be: $l_{dis} : Z_m \longrightarrow L_{dis}$ where $Z_m = \{0, 1, ..., m - 1\}$ and L_{dis} is a set of distance relation labels in natural language (e.g. $\{veryclose, close, neural, far, veryfar\}$). Its converse function is defined as $l_{dis}^{-1} : L_{dis} \longrightarrow Z_m$. In this paper the default qualitative distance calculus used is $\mathcal{QDR}_5$ and the default l_{dis} function (as in Figure 3a) is

- $l_{dis}(0) = very\_close$
- $l_{dis}(1) = close$
- $l_{dis}(2) = neural$
- $l_{dis}(3) = far$
- $l_{dis}(4) = very\_far$

The value of m and the corresponding l_{dis} function can also be configured by users because in different cases the number of regions, region widths and the natural language terms to describe each region might be also different. For example, people might just use "close" and "far" (i.e. $m = 2$ as in Figure 3b) to describe distance relations rather than the more refined 5 terms used in the default l_{dis} function. Also, the range of "near" varies in different contexts [27].

3.3 Extracting Qualitative Knowledge from Natural Language Descriptions

It is possible to extract direction relations from route descriptions by applying natural language processing techniques discussed in [21]. Such qualitative knowledge extraction is more related to Natural Language Processing and is not the main focus of this paper. We just assume qualitative knowledge can be extracted from natural language descriptions. For example, the direction relations in "going from A to B you can see C is on the left" and "going from D to B you can see C is on the right" can be denoted as $r_{dir}(A, B, C) = l_{dir}^{-1}(left)$ and $r_{dir}(D, B, C) = l_{dir}^{-1}(right)$ respectively. Since there might be different incoming places (e.g. A and D), the deciding orientation of the direction relation between two places might also be multiple. If the incoming place is unknown as in the description "B is on A's leftside", the direction relation between A and B has to be denoted as $r_{dir}(null, A, B) = l_{dir}^{-1}(left)$, which actually provides no information regarding the relative position of B to A if no further information is given.

It is also possible to extract distance relations, reachability relations and visibility relations from natural language descriptions. For example, the distance relation in "A is very close to B" can be denoted as $r_{dis}(A, B) = l_{dis}^{-1}(very\_close)$; the reachability relation in "from A you can go to B"can be denoted as $r_{rea}(A, B) = t$; and the visibility relation in "at A you can see B" can be denoted as $r_{vis}(A, B) = t$.

3.4 Map Construction

Given the qualitative knowledge of an environment in the form of a set of direction relations, distance relations, reachability relations and visibility relations, we can construct a qualitative place map by applying Algorithm 1.

Before iterating those qualitative relations, we first infer new relations by the following rules and add them to the corresponding sets of relations (Line 2):

- If $r_{dir}(A, B, C) = i$, $r_{dir}(C, B, A) = 4m - i$, which is based on the converse rules discussed in [16].
- If $r_{dis}(A, B) = i$, $r_{dis}(B, A) = i$. It is based on the assumption that the distance from A to B is the same as that from B to A.
- If $r_{rea}(A, B) = t$, $r_{rea}(B, A) = t$, which is based on the assumption that if B is reachable from A then A is also reachable from B.
- If $r_{rea}(A, B) = t, r_{vis}(A, B) = t$. It is assumed if B is reachable from A then it is also visible from A.
- If $r_{vis}(A, B) = t$, $r_{vis}(B, A) = t$, which is based on the assumption that if B is visible from A then A is also visible from B.

Note that most of the above rules are based on simplified assumptions, which is subject to change in different scenarios. For example, it is possible that Place A is directly reachable from Place B but not visible from Place B because they are connected by a curved path and there is an obstacle inbetween so that Place A is not visible from Place B. Other qualitative spatial reasoning techniques (e.g. composition) can also be used to infer new qualitative relations. However, it is still an open problem whether they are that helpful in our case. For example, the direction relation between Place A, B and D (i.e. $r_{dir}(A, B, D)$) can be inferred by composing $r_{dir}(A, B, C)$ and $r_{dir}(B, C, D)$. But it is possible that D is neither visible nor not directly reachable from B, which means the newly inferred direction relation between A, B, and D may not be helpful in localization or navigation.

Then we iterate the set of direction relations S_{dir} (Line 4). Given a direction relation between three places for example $r_{dir}(p_1, p_2, p_3) = i$ and $i \in Z_{4m}$ (Line 5), check whether p_1, p_2, and p_3 have already been represented by vertices in V (Line 6) and the edge from the vertex of p_2 to the vertex of p_3 is in E (Line 7). If the name of a place is semantic (Line 27), we can check whether there is a vertex in V that has the same place name (if yes it is already in V); if the name of a place is functional (Line 32), we just add it to V because so far it is impossible to judge for example whether two intersections are actually referring to the same place. In the QPM, the direction relation between v_2 and v_3 is denoted as (v_1, i) and it will be added as one of the direction relations between v_2 and v_3 in the label of the edge from v_2 to v_3 (Line 8) where v_1, v_2, v_3 are the vertices representing p_1, p_2, and p_3 respectively.

Similarly, next we go through the set of distance relations S_{dis} (Line 9). Given a distance relation between two places for example $r_{dis}(p_1, p_2) = i$ and $i \in Z_m$ (Line 10), check whether these two places and the edge between them are already in the QPM (Line 11-12). Then add (i) as the distance relation in the label of the edge from v_1 to v_2 (Line 13). The same procedure is applied in iterating through the set of reachability relations S_{rea} (Line 14-18) and visibility relations S_{vis} (Line 19-23).

Note that in a qualitative place map, if no other information is given, the default direction relation between two vertices is $U_{dir} = \{0, 1, ..., 15\}$ (namely can be any relation in this set); the default

Algorithm 1 Qualitative Place Map Construction

1: **procedure** CONSTRUCT($QPM, S_{dir}, S_{dis}, S_{rea}, S_{vis}$)
2: InferAndAddNewRelation($S_{dir}, S_{dis}, S_{rea}, S_{vis}$)
3: QPM = (V, E)
4: **for** $r \in S_{dir}$ **do**
5: $r = (p_1, p_2, p_3, i)$
6: CheckPlace(r, V)
7: e = CheckEdge(p_2, p_3, E)
8: $e.R_{dir}.add((v_1, i))$
9: **for** $r \in S_{dis}$ **do**
10: $r = (p_1, p_2, i)$
11: CheckPlace(r, V)
12: e = CheckEdge(p_1, p_2, E)
13: $e.r_{dis} = i$
14: **for** $r \in S_{rea}$ **do**
15: $r = (p_1, p_2, t)$
16: CheckPlace(r, V)
17: e = CheckEdge(p_1, p_2, E)
18: $e.r_{rea} = t$
19: **for** $r \in S_{vis}$ **do**
20: $r = (p_1, p_2, t)$
21: CheckPlace(r, V)
22: e = CheckEdge(p_1, p_2, E)
23: $e.r_{vis} = t$
24:
25: **procedure** CHECKPLACE(r, V)
26: **for** $p \in r.places$ **do**
27: **if** $p.name.type = semantic$ **then**
28: **if** $p.name \notin V.placeNames$ **then**
29: $pn(v) = p.name$
30: $lm(v) = t$
31: $V.add(v)$
32: **if** $p.name.type = functional$ **then**
33: $pn(v) = p.name$
34: $lm(v) = f$
35: $V.add(v)$

distance relation is 2; both of the default reachability relation and visibility relation are f.

Algorithm 2 Qualitative Place Map Construction by Exploration

```
procedure Construct(V, E, p)
    p.explored = true
    AddVertice(p, V)
    for p_nb ∈ p.neighbors do
        if p_nb.explored ≠ true then
            AddEdge(p, p_nb, E)
            RecordRelations()
            Construct(V, E, p_nb)
            AddEdge(p_nb, p, E)
            RecordRelations()
```

A qualitative place map can also be constructed by exploration if there is a traveller or a robot that can move around and record observations. The exploration strategy can be based on a depth-first search (DFS) [32] so that we can traverse all places in a certain order. Start from a random place, we can mark it as "explored" and then move to an arbitrary next place that is directly reachable. If the next place has already been marked as "explored", go back to the incoming place and choose the next directly reachable place. Record all of the observations about the direction relations, distance relations, reachability relations and visibility relations between places during the exploration. The exploration process can stop if at a certain place all of the directed reachable places have been explored. Algorithm 2 describes this process of starting from a random place p in a place network and the QPM constructed is represented by (V, E).

4 INTERACTIVE LOCALIZATION

Localization is important because the starting position of an agent might be unknown, or it might be lost during the navigation process. It is trivial to locate a user if GPS is available while there are still many GPS-denied environments. In these cases our localization mechanism is still available given the prelearned qualitative place map (QPM) of the environment.

Our localization algorithm (Algorithm 3) is interactive because it is based on the interaction with the agent. Details are as follows:

(1) Initialize a candidate set V_c that includes all landmarks in the given QPM (Line 3-5).

(2) The agent is asked to move to an arbitrary next place and to provide descriptions about its current position (Line 7): it can go to the nearest salient place and tell us the name of the place (*place*); or it can describe the places that are visible from its current position (S_{vis}); or it can tell us from which incoming landmark the direction relations between its current place and other visible landmarks (S_{dir}). We assume in each communication only one form of descriptions is provided by the agent.

(3) If the information given by the agent is the name of its current place (Line 11), we check whether it is included as a landmark in the QPM. If yes then we know the current position of the agent and let V_c only contain this landmark (Line 13-16).

(4) If the information given by the agent is the names of the places that are visible from its current position (Line 17), we can remove

Algorithm 3 Interactive Localization

```
1:  procedure InteractiveLocalization(QPM)
2:      QPM = (V, E)
3:      for v ∈ V do
4:          if lm(v) = t then
5:              V_c.add(v)
6:      while |V_c| > 1 do
7:          (place, S_vis, S_dir) = AgentDescription()
8:          Locate(place, S_vis, S_dir, V, E, V_c)
        return V_c
9:
10: procedure Locate(place, S_vis, S_dir, V, E, V_c)
11:     if place ≠ ∅ then
12:         for v ∈ V do
13:             if pn(v) = place.name then
14:                 V_c.removeAll()
15:                 V_c.add(v)
16:                 return
17:     if S_vis ≠ ∅ then
18:         for v ∈ V_c do
19:             isValid = true
20:             for r ∈ S_vis do
21:                 r = (p_1, p_2, t)
22:                 if r_vis(v, v(p_2)) = f then
23:                     isValid = false
24:             if isValid = false then
25:                 V_c.remove(v)
26:     if S_dir ≠ ∅ then
27:         for v ∈ V_c do
28:             isValid = true
29:             for r ∈ S_dir do
30:                 r = (p_1, p_2, p_3, i)
31:                 if r_dir(v(p_1), v, v(p_3)) ≠ i then
32:                     isValid = false
33:             if isValid = false then
34:                 V_c.remove(v)
```

landmarks from V_c if from it any of the places mentioned are not visible (Line 22). $v(p_2)$ is the vertex that represents p_2. $r_{vis}(v_1, v_2)$ returns the visibility relation between v_1 and v_2.

(5) If the information given by the agent is the direction relations between its current position and other landmarks from a certain incoming landmark (Line 26), we can remove landmarks from V_c if from the incoming landmark the direction relations between it and the landmarks do not match the direction relations provided by the agent (Line 31). $v(p_1)$ is the vertex that represents p_1 and $v(p_3)$ is the vertex that represents p_3. $r_{dir}(v_1, v_2, v_3)$ returns the direction relation between v_2 and v_3 given v_1 as the incoming landmark.

(6) If the number of elements in V_c is greater than 1, go back to step (2) (Line 6).

It is assumed that the orientation of an agent is the same as the orientation from the incoming place to its current place. The incoming landmark must be included in a direction relation description because, as discussed above, if it is not provided the direction relation given is informationless. Our localization algorithm can always succeed to locate an agent if the information provided by the agent is consistent and the agent is not allowed to keep moving in a loop of places (which is supported by the experimental results in Section 7). If the current location of the agent cannot be determined in one query, the agent has to requery. The number of requeries for localization (i.e. the number of interactions with the system minus 1) should not be greater than a certain value (e.g. 2), otherwise the agent will be frustrated. In Section 7 we will explore whether our localization algorithm can always succeed in agent localization and also the average number of requeries.

5 RELIABLE NAVIGATION

5.1 Route Planning

Route planning aims to find the "best" route from the origin to the destination though "best" is defined by different criteria in different route planning algorithms. For example, the classical Dijkstra algorithm [7] is to find the route with the shortest length while the shortest most reliable path algorithm [14] (which is adaptation of the Dijkstra algorithm) aims to find the most reliable route and route length is used to break ties when two routes share the same unreliability. These two algorithms are also different in number of nodes involved in each assignment of the weighting function (which is designed to assign weight to each edge or a pair of edges): 2 in the Dijkstra algorithm because the weighting function is only related to one edge; 3 in the shortest most reliable path algorithm because the main weighting function is related to two edges which satisfies that ending node of the first edge is the same as the starting node of the second edge. To avoid the repetitive description on how the Dijkstra algorithm is adapted, we just focus on how to calculate the total cost of a route. The cost function is defined to assign the cost to a route and it indicates the cost of moving from the first node to the last node by making turns or going straight via the intermediate nodes. For example, the cost function used in [14] is:

$$cost[v_1, v_2, ..., v_n] = \sum_{i=1}^{n-2}(ie[v_i, v_{i+1}, v_{i+2}] - 1) \tag{1}$$

where $cost[v_1, v_2, ..., v_n]$ is the cost of a route from v_1 to v_n and $ie[v_i, v_{i+1}, v_{i+2}]$ is the number of turns that are instruction equivalent to the turn from the edge (v_i, v_{i+1}) to the edge (v_{i+1}, v_{i+2}) (or more formally the number of vertices v that satisfy

$$r_{dir}(v_i, v_{i+1}, v) = r_{dir}(v_i, v_{i+1}, v_{i+2})). \tag{2}$$

In contrast, the cost function used in our route planning algorithm $cost^*[v_1, v_2, ..., v_n] =$

$$\sum_{i=1}^{n-2}(\frac{log_{a_1}(ie[v_i, v_{i+1}, v_{i+2}] - 1 + a_1)}{log_{a_2}(lm_\#[v_{i+2}] + a_2)}) + qd[v_1, v_2, ..., v_n] \tag{3}$$

where $lm_\#[v_{i+2}]$ is the number of landmarks that are visible at v_{i+2} and $qd[v_1, v_2, ..., v_n]$ is the qualitative distance from v_1 to v_n (or in a more formal form

$$qd[v_1, v_2, ..., v_n] = \sum_{i=1}^{n-1}(r_{dis}(v_i, v_{i+1}))). \tag{4}$$

Our route planning algorithm considers not only the distance between two nodes and the unreliability of each turn but also the number of landmarks involved in a certain route because it is less possible for an agent to be lost and the route instructions can be simpler with greater number of landmarks. Its cost function consists of two parts: the first part is the sum of the ratios of the unreliablity of the turns to the number of landmarks. For each turn, it is better to calculate the ratio of these two factors because they might be relevant (e.g. greater number of landmarks might incur the increase of the unreliablity as in Figure 4). Note that logarithmic functions are applied to smooth the influence of much greater values of the unreliability of a turn and the number of landmarks. a_1 and a_2 are adjustable constants and their values can be decided by exploratory experiments as discussed in Section 7. The second part is the qualitative distance from the origin to the destination.

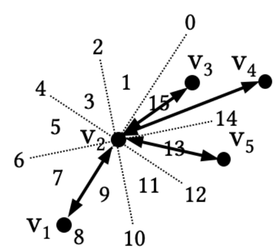

Figure 4: Assuming v_4 is a landmark and is visible from v_3. When moving from v_1 to v_3 via v_2, v_4 is helpful as a landmark that might be used to locate v_3. However, v_4 can also increase the unreliability because it is the turn of (v_1, v_2, v_3) is instruction equivalent to the turn of (v_1, v_2, v_4).

Other details of our route planning algorithm are omitted because the only difference between our algorithm and the shortest most reliable path algorithm [14] is how the cost function is defined. The discussion about the computational complexity is also similar (namely $O(|E|^2)$).

Algorithm 4 Reliable Navigation

1: **procedure** NAVIGATION(QPM, v_d)
2: $\quad v_c = InteractiveLocalization(QPM)$
3: $\quad route = FindTheMostReliableRoute(QPM, v_c, v_d)$
4: $\quad$ **for** $(v_1, v_2) \in route$ **do**
5: $\quad\quad GenerateInstructions(v_1, v_2)$

5.2 Navigation System

As in Algorithm 4, by applying the interactive localization mechanism discussed in Section 4 our navigation algorithm can navigate an agent without knowing its current place (Line 2).

Given the qualitative place map, the current vertex v_c and the destination vertex v_d, the route planning algorithm discussed in the previous subsection can be used to find the most reliable route (Line 3).

Extract the edges in the most reliable route in sequence and generate route instructions (Line 5). A template-based route generation strategy is used in our system while more sophisticated strategies can be found in [6, 24]. For example, given v_1 as the current place and v_2 the next place to go, first describe landmarks that are visible from v_1 in the form of "at $pn(v_1)$ you can see..." and then indicate whether to go straight or take turns to get to v_2 based on the direction relation between v_1 and v_2; next describe landmarks that are visible from v_2 by using the same template "at $pn(v_2)$ you can see...". This mechanism is to ensure that each time the agent arrives at a new place, descriptions of visible landmarks from the new place will be provided to check whether the agent is at the correct place, which is better than [14, 34] where an agent can only realize they are in the wrong way when there is no valid navigation instruction to follow.

Whenever the agent reports it is lost, just restart the navigation procedure.

6 CASE STUDY

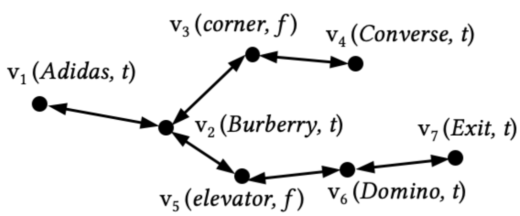

Figure 5: A qualitative place map constructed from two route descriptions.

In this section we will describe how to solve the problem mentioned in the introduction by using our navigation system. As shown in Figure 5 a qualitative place map can be constructed based on the qualitative knowledge extracted from the two descriptions and applying the inference rules discussed in Section 3.4. In the QPM, each vertex is labelled with a tuple where the first element is the name of the place and the second element indicates whether it is a landmark. The labels of edges are as follows:

- $e_{12} = (v_1, v_2, t, t, \{(null, l_{dir}^{-1}(left))\}, 2)$
- $e_{21} = (v_2, v_1, t, t, \{(null, U_{dir}\}, 2)$
- $e_{23} = (v_2, v_3, t, t, \{(v_1, U_{dir})\}, 1)$
- $e_{32} = (v_3, v_2, t, t, \{(v_4, l_{dir}^{-1}(right))\}, 1)$
- $e_{34} = (v_3, v_4, t, t, \{(v_2, l_{dir}^{-1}(left))\}, 3)$
- $e_{43} = (v_4, v_3, t, t, \{(null, U_{dir})\}, 3)$
- $e_{25} = (v_2, v_5, t, t, \{(null, U_{dir})\}, 0)$
- $e_{52} = (v_5, v_2, t, t, \{(v_6, l_{dir}^{-1}(left))\}, 0)$
- $e_{56} = (v_5, v_6, t, t, \{(v_2, l_{dir}^{-1}(right))\}, 1)$
- $e_{65} = (v_6, v_5, t, t, \{(v_7, l_{dir}^{-1}(left))\}, 1)$
- $e_{67} = (v_6, v_7, t, t, \{(v_5, l_{dir}^{-1}(right))\}, 0)$
- $e_{76} = (v_6, v_5, t, t, \{(null, U_{dir})\}, 0)$

Applying the routing planning algorithm discussed in Section 5, it is not difficult to know that the best route from "Converse" (v_4) to "Exit" (v_7) is $v_4 v_3 v_2 v_5 v_6 v_7$. An agent can follow the route instructions generated by the template-based strategy discussed in Section 5.2:

- From Converse go to the corner and turn right to Burberry. At the corner you can see Burberry.
- At Burberry you can see Adidas. From Burberry go to the elevator and turn right to Domino. At the elevator you can see Domino.
- From Domino you can see Exit. From Domino go to Exit.

It is not difficult to recognize the templates used in the above route instructions. Generating more human-like instructions are left as future work.

7 EXPERIMENTS

We evaluate the whole localization and navigation process in simulated environments. Qualitative place maps (QPMs) are generated by the Exploration Algorithm (Algorithm 2) from graphs of places with different configurations in terms of number of vertices and edges. Specifically, the four configurations being used are 200 vertices & 200 edges, 200 vertices & 300 edges, 500 vertices & 500 edges and 500 vertices & 750 edges. The vertices may be recognizable landmarks or non-salient places. The landmark vertices may be reachable by the agent (e.g. a bookshop) or only visible but unreachable (e.g. an advertisement board hanging on the ceiling). The qualitative place maps are built with the default qualitative spatial models used in this paper: $\mathcal{RDR}_4$ and $\mathcal{QDR}_5$.

Then, we simulate the guiding process between a navigator with the qualitative place maps and an agent who aims to travel from randomly selected starting points to destination points, which corresponds to the situation where based on prior knowledge one person can localize and navigate the other via remote communications. Note that it is assumed the starting point is unknown to both the navigator and the agent at first. The navigator needs to locate the agent based on its qualitative place map and the observations provided by the agent. Another assumption we made here is that the agent will correctly interpret and execute route instructions, which means, the agent can only make mistakes when facing with ambiguous turns.

7.1 Evaluation of Interactive Localization

In this subsection, our interactive localization algorithm is evaluated by analyzing the number of requeries needed for obtaining the correct location of the agent. In our localization mechanism, each requery results in a movement of the agent, therefore it is important to keep the number of requeries small. Table 1 demonstrates the percentage of requery amounts performed over 10000 testing cases in each place graph. The accuracy of our qualitative localization algorithm is 100% (i.e. the agent can always be located). More than 80% of the localization processes need zero or one requery, and more than 90% of the localization processes need no more than 2 requeries. Based on the experimental results, it is safe to claim our method works reasonably well though there is no similar purely qualitative localization algorithm to compare with.

It can also be observed from the results that with the increase of number of vertices and edges, the percentage of tests with more than 1 requeries slightly increases. One possible reason is that more requeries are required to resolve the ambiguity brought by the increased vertices and their connections.

Table 1: Evaluation results of interactive localization. 'v' represents the number of vertices in the place graph; 'e' represents the number of edges in the place graph; 'r' is the number of requeries; 'r_{ave}' is the average number of requeries made per guidance.

	v=200,e=200	v=200,e=300	v=500,e=500	v=500,e=750
$r = 0$	38.01%	37.32%	34.19%	34.31%
$r = 1$	49.13%	49.05%	52.12%	47.60%
$r = 2$	6.49%	8.03%	7.14%	10.53%
$r > 2$	6.37%	5.60%	6.55%	7.56%
r_{ave}	0.90	0.87	0.92	0.99

7.2 Evaluation of Reliable Navigation

In this subsection, we evaluate the navigation process which includes interactive localization and route planning. We monitor the number of instructions given by the navigator (namely the number of places from the origin to the destination), the distance that the agent actually travels in the simulation and the number of wrong turns the agent makes during the travel. We compare our results with the Dijkstra algorithm and also the shortest most reliable path algorithm (SMR) described in [14]. Since it is assumed that the exact distances between places are unknown to the navigator, the distances used in the three algorithms are all qualitative distances with granularity of 5. However, the total travel distance for measuring the performance of the three algorithms are exact distances from the ground truth of the place graphs. In the evaluation below, the two constants in our cost function (i.e. a_1 and a_2 in Equation (3)) are set to 4 and 2 respectively based on the exploratory experiments on our algorithm.

Dijkstra, SMR, and our interactive localization and reliable navigation (ILRN) algorithm are all applied in 10000 randomly generated test cases in each of the same four graphs (as those in Section 7.1). The average number of instruction, average number of wrong turns and average distance are calculated for comparison. As shown in Table 2, 3, 4 and 5, we can observe that our algorithm provided the smallest number of instructions among the three algorithms in all four different configurations. By using our algorithm the average number of wrong turns is much smaller than those by using the other two algorithms, which demonstrates our algorithm is capable of finding the most reliable route and can reduce the frequency of getting lost. Note that in two situations wrong turns will be detected: 1) descriptions from the agent are different from the expectation of the navigator; 2) the agent cannot observe the landmark according to the route instruction given by the navigator.

For the relatively large graphs with 500 vertices, by using our algorithm the smallest travel distance can be achieved. However, for graphs with 200 vertices, the physically shortest paths cannot be found by using our algorithm. But as discussed above, our method is still preferable when compared to the other two algorithms because it can largely reduce the number of instructions from the navigator and the number of wrong turns.

Table 2: Evaluation results for different cost functions on a place graph with 200 vertices and 200 edges. "*#instructions*" is the number of instructions provided by the navigator to the agent. "*#wrong turns*" is the number of incorrect turns made by the agent because of ambiguous turns. "*distance*" is the travelling distance (in units of length) of the agent including wrong movements and movements by the interactive localization mechanism. All the data in the following 4 tables are averaged per trip. "SMR" is short for the shortest most reliable path algorithm and "ILRN" is short for our interactive localization and reliable navigation algorithm.

	Dijkstra	SMR	ILRN
# instructions	9.27	9.14	8.90
# wrong turns	0.75	0.72	0.61
distance	4.03	4.03	4.10

Table 3: Evaluation results for different weight configurations on a place graph with 200 vertices and 300 edges.

	Dijkstra	SMR	ILRN
# instructions	7.99	7.94	7.68
# wrong turns	1.00	0.93	0.71
distance	3.53	3.43	3.54

Table 4: Evaluation results for different weight configurations on a place graph with 500 vertices and 500 edges.

	Dijkstra	SMR	ILRN
# instructions	11.74	11.60	11.28
# wrong turns	0.76	0.71	0.61
distance	5.23	5.18	5.12

Table 5: Evaluation results for different weight configurations on a place graph with 500 vertices and 750 edges.

	Dijkstra	SMR	ILRN
# instructions	10.55	10.34	9.54
# wrong turns	1.42	1.30	0.95
distance	4.86	4.72	4.50

8 CONCLUSION AND FUTURE WORK

In this paper we have provided a series of mechanisms that enable localization and navigation in GPS-denied environments given only qualitative knowledge of places. This is motivated by human communication and aims to assist human localization and navigation in an intuitive and interactive way. The qualitative knowledge between places (landmarks or not) is extracted from natural language descriptions and stored in a directed graph called qualitative place map where the vertices represent places and edges are labelled with different categories of qualitative relations between places that might be helpful in localization and navigation. Based on qualitative place maps, an interactive localization algorithm has been described that can locate agents based on eliminating landmarks that cannot satisfy the observations provided by agents;

and a reliable navigation system has been introduced which (different from previous methods) also takes number of landmarks and qualitative distances into consideration. In the experiments, we demonstrate the usefulness of our localization algorithm as well as the improvement in reliable route calculation compared to previous navigation systems. More importantly, those previous navigation systems might not be useful in cases where the coordinates of places are not given. With our qualitative localization algorithm, some previous GPS-based navigation systems can also be used in GPS-denied environments by performing localization based on the observations from the agent or user.

In the future, we plan to combine our system with more sophisticated natural language processing techniques and extend its practical usefulness by applying it in more real-world scenarios.

9 ACKNOWLEDGMENTS

This research is funded by ARC DP170100109.

REFERENCES

[1] Federico Boniardi, Bahram Behzadian, Wolfram Burgard, and Gian Diego Tipaldi. 2015. Robot navigation in hand-drawn sketched maps. In *2015 European conference on mobile robots (ECMR)*. IEEE, 1–6.

[2] Hao Chen, Maria Vasardani, Stephan Winter, and Martin Tomko. 2018. A Graph Database Model for Knowledge Extracted from Place Descriptions. *ISPRS International Journal of Geo-Information* 7, 6 (2018), 221.

[3] George Chronis and Marjorie Skubic. 2004. Robot navigation using qualitative landmark states from sketched route maps. In *IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA'04. 2004*, Vol. 2. IEEE, 1530–1535.

[4] Anthony G. Cohn and Jochen Renz. 2008. Qualitative Spatial Representation and Reasoning. In *Handbook of Knowledge Representation*. Elsevier, 551–596.

[5] Marie-Paule Daniel and Michel Denis. 1998. Spatial descriptions as navigational aids: A cognitive analysis of route directions. *Kognitionswissenschaft* 7, 1 (1998), 45–52.

[6] Andrea F Daniele, Mohit Bansal, and Matthew R Walter. 2017. Navigational instruction generation as inverse reinforcement learning with neural machine translation. In *2017 12th ACM/IEEE International Conference on Human-Robot Interaction (HRI*. IEEE, 109–118.

[7] Edsger W Dijkstra. 1959. A note on two problems in connexion with graphs. *Numerische mathematik* 1, 1 (1959), 269–271.

[8] Matt Duckham and Lars Kulik. 2003. "Simplest" paths: automated route selection for navigation. In *International Conference on Spatial Information Theory*. Springer, 169–185.

[9] Matt Duckham, Stephan Winter, and Michelle Robinson. 2010. Including landmarks in routing instructions. *Journal of Location Based Services* 4, 1 (2010), 28–52.

[10] Frank Dylla. 2009. Qualitative spatial reasoning for navigating agents. *Behaviour Monitoring and Interpretation: Ambient Assisted Living, B. Gottfried and H. Aghajan, Eds. IOS Press* (2009), 98–128.

[11] Irene Fellner, Haosheng Huang, and Georg Gartner. 2017. "Turn Left after the WC, and Use the Lift to Go to the 2nd Floor" Generation of Landmark-Based Route Instructions for Indoor Navigation. *ISPRS International Journal of Geo-Information* 6, 6 (2017), 183.

[12] Paolo Fogliaroni, Jan Oliver Wallgrün, Eliseo Clementini, Francesco Tarquini, and Diedrich Wolter. 2009. A qualitative approach to localization and navigation based on visibility information. In *International Conference on Spatial Information Theory*. Springer, 312–329.

[13] Christian Freksa. 1992. Using orientation information for qualitative spatial reasoning. In *Theories and methods of spatio-temporal reasoning in geographic space*. Springer, 162–178.

[14] Shazia Haque, Lars Kulik, and Alexander Klippel. 2006. Algorithms for reliable navigation and wayfinding. In *International Conference on Spatial Cognition*. Springer, 308–326.

[15] Daniel Hernandez, Eliseo Clementini, and Paolino Di Felice. 1995. Qualitative distances. In *International Conference on Spatial Information Theory*. Springer, 45–57.

[16] Hua Hua, Jochen Renz, and Xiaoyu Ge. 2018. Qualitative Representation and Reasoning over Direction Relations across Different Frames of Reference. In *KR 2018*. 551–560.

[17] Alexander Klippel, Heike Tappe, Lars Kulik, and Paul U Lee. 2005. Wayfinding choremes a language for modeling conceptual route knowledge. *Journal of Visual Languages & Computing* 16, 4 (2005), 311–329.

[18] Bernd Krieg-Brückner and Hui Shi. 2006. Orientation calculi and route graphs: Towards semantic representations for route descriptions. In *International Conference on Geographic Information Science*. Springer, 234–250.

[19] Theocharis Kyriacou, Guido Bugmann, and Stanislao Lauria. 2005. Vision-based urban navigation procedures for verbally instructed robots. *Robotics and Autonomous Systems* 51, 1 (2005), 69–80.

[20] Tod S Levitt. 1990. Qualitative navigation for mobile robots. *Int. J. Artificial Intelligence* 44 (1990), 305–360.

[21] Fei Liu, Maria Vasardani, and Timothy Baldwin. 2014. Automatic identification of locative expressions from social media text: A comparative analysis. In *Proceedings of the 4th International Workshop on Location and the Web*. ACM, 9–16.

[22] Dominik Lücke, Till Mossakowski, and Reinhard Moratz. 2011. Streets to the OPRA Finding your destination with imprecise knowledge. In *Proc. IJCAI Workshop on Benchmarks and Applications of Spatial Reasoning*. Citeseer, 25–32.

[23] Pierre-Emmanuel Michon and Michel Denis. 2001. When and why are visual landmarks used in giving directions?. In *International Conference on Spatial Information Theory*. Springer, 292–305.

[24] Stefan Oßwald, Henrik Kretzschmar, Wolfram Burgard, and Cyrill Stachniss. 2014. Learning to give route directions from human demonstrations. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 3303–3308.

[25] Jochen Renz and Debasis Mitra. 2004. Qualitative direction calculi with arbitrary granularity. In *PRICAI*, Vol. 3157. 65–74.

[26] Jochen Renz and Bernhard Nebel. 2007. Qualitative Spatial Reasoning Using Constraint Calculi. In *Handbook of Spatial Logics*. Springer, 161–215.

[27] Daniela Richter, Stephan Winter, Kai-Florian Richter, and Lesley Stirling. 2013. Granularity of locations referred to by place descriptions. *Computers, Environment and Urban Systems* 41 (2013), 88–99.

[28] Kai-Florian Richter and Matt Duckham. 2008. Simplest instructions: Finding easy-to-describe routes for navigation. In *International Conference on Geographic Information Science*. Springer, 274–289.

[29] Davide Russo, Sisi Zlatanova, and Eliseo Clementini. 2014. Route directions generation using visible landmarks. In *Proceedings of the sixth ACM SIGSPATIAL international workshop on indoor spatial awareness*. ACM, 1–8.

[30] M Serrão, João MF Rodrigues, and JM Hans du Buf. 2014. Navigation framework using visual landmarks and a GIS. *Procedia Computer Science* 27 (2014), 28–37.

[31] Danelle C Shah and Mark E Campbell. 2013. A qualitative path planner for robot navigation using human-provided maps. *The International Journal of Robotics Research* 32, 13 (2013), 1517–1535.

[32] Robert Tarjan. 1972. Depth-first search and linear graph algorithms. *SIAM journal on computing* 1, 2 (1972), 146–160.

[33] Maria Vasardani, Sabine Timpf, Stephan Winter, and Martin Tomko. 2013. From descriptions to depictions: A conceptual framework. In *International Conference on Spatial Information Theory*. Springer, 299–319.

[34] Matthias Westphal and Jochen Renz. 2011. Evaluating and minimizing ambiguities in qualitative route instructions. In *Proceedings of the 19th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*. ACM, 171–180.

[35] Gangani Geethika Wijewardena, Maria Vasardani, and Stephan Winter. 2016. Towards Indoor Localization and Navigation Independent of Sensor Based Technologies. In *Eighth ACM SIGSPATIAL International Workshop on Indoor Spatial Awareness*, Muhammad Aamir Cheema and Mohammed Eunus Ali (Eds.). ACM Press.

[36] Stephan Winter. 2002. Modeling costs of turns in route planning. *GeoInformatica* 6, 4 (2002), 345–361.

[37] Stephan Winter, Kai-Florian Richter, Mingzheng Shi, and Heng-Soon Gan. 2011. Get me out of here: Collaborative evacuation based on local knowledge. In *Proceedings of the 3rd ACM SIGSPATIAL International Workshop on Indoor Spatial Awareness*. ACM, 35–42.

[38] Stephan Winter, Martin Tomko, Maria Vasardani, Kai-Florian Richter, Kourosh Khoshelham, and Mohsen Kalantari. 2019. Infrastructure-Independent Indoor Localization and Navigation. *Comput. Surveys* 52, 3 (2019), 61:1–61:24.