Incorporating Compositions in Qualitative Approaches

Itai Zilberman[1], Ehud Rivlin[2] and Vadim Indelman[3]

***Abstract*—Qualitative approaches to various tasks, ranging from localization and mapping to active planning, are gaining considerable momentum in recent years. These approaches represent the environment through spatial relationships between small sets of landmarks in independent local coordinate systems. An essential component in these approaches is the composition operator, enabling spatial information propagation between different sets to infer new ones. Integrating compositions within qualitative algorithms brings several difficulties. For instance, if the information required to perform a specific composition operation is unavailable, it must be inferred first, possibly via a preparatory composition operation. This recursive issue becomes more challenging as the amount of information grows. This paper addresses two main questions arising from the above, which remained open: 1. Given an initial set of qualitative spatial relationships, what new ones can be composed? 2. What is the optimal sequence of compositions operations to create a target set among all possible sequences? We provide a theoretical derivation to address the first question and a novel search algorithm to address the second.**

***Index Terms*—Autonomous Agents, Localization, Mapping.**

I. Introduction

MANY robotics applications rely on accurate metric estimations of the environment and robot's location to accomplish their aims. However, in the absence of high-quality sensors, achieving high accuracy can be very challenging.

While maintaining accurate information is often essential, it might be unnecessary in some cases. For instance, consider an autonomous cleaning robot operating in a living room, as illustrated in Fig. 1a. A typical living room contains a relatively small number of meaningful objects. Relying on rough relative relationships between the different objects, rather than on exact metric coordinates, may be sufficient for the robot to maneuver within the room successfully. E.g., if the robot seeks to clean under the table, it must pass safely between the table's legs. However, neither the exact metric coordinates of the legs nor the exact robot's location between them is required.

Manuscript received: September, 6, 2021; Revised November, 3, 2021; Accepted December, 31, 2021. This paper was recommended for publication by Editor Markus Vincze upon evaluation of the Associate Editor and Reviewers' comments. This work was partially supported by the Israel Ministry of Science & Technology (MOST).

[1]Itai Zilberman is with the Department of Electrical Engineering, Technion - Israel Institute of Technology, Haifa 32000, Israel silber@campus.technion.ac.il

[2]Ehud Rivlin is with the Department of Computer Science, Technion - Israel Institute of Technology, Haifa 32000, Israel ehudr@cs.technion.ac.il

[3]Vadim Indelman is with the Department of Aerospace Engineering, Technion - Israel Institute of Technology, Haifa 32000, Israel vadim.indelman@technion.ac.il

Digital Object Identifier (DOI): see top of this page.

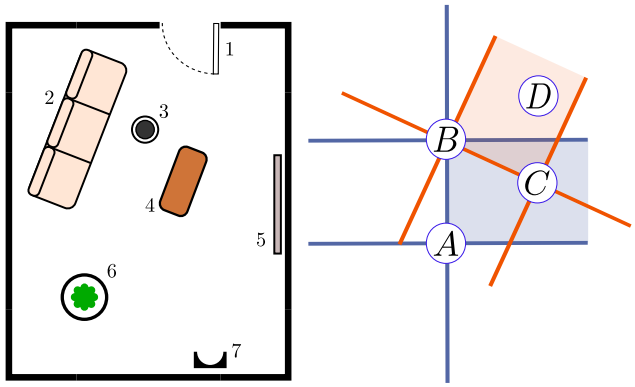

Figure 1: (a) A simplified living room scene. 1-door; 2-sofa; 3-cleaning robot; 4-table; 5-TV; 6-flowerpot; 7-charger. Given known QSRs, the cleaning robot can compose new ones, which may aid it in accomplishing its task; (b) Illustration of the composition operation. Given evaluated qualitative states for $AB{:}C$ ("Middle Right") and $BC{:}D$ ("Middle Left"), the composition operator determines which qualitative states for $AB{:}D$ are feasible ("Top Right" and "Middle Right"). In practice, any ordered combination of ABD (i.e., $AB{:}D$, $AD{:}B$, $BA{:}D$, $BD{:}A$, $DA{:}B$, $DB{:}A$) can be composed using any ordered combination of ABC and BCD (with additional unary operations in some implementations). Thus, in the paper, we omit the ordered notation of $':'$.

Qualitative approaches are motivated by the above. In contrast to the metrical methods, the environment and robot's poses are tracked using coarse, relative geometrical relations, known as qualitative spatial relationships (QSRs). Each QSR fixes a coordinate system based on a small set of landmarks and discretizes space into disjoint regions, called qualitative states (see Fig. 2). Then, the location of a target landmark or robot pose is described in terms of these states.

Since qualitative approaches represent spatial information through a set of QSRs, each in its own local independent coordinate frame, propagating information between different relationships may be essential. The composition operator was first introduced in 1992 out of this very need [1]. Given a pair of source QSRs, the latter aims to conclude the third one under some topological conditions. For example, consider a scenario where the above-mentioned cleaning robot has finished its work and is currently located between the sofa and the table. Suppose the robot seeks to return to its charger. Consider τ_1 to be a known QSR that describes the flowerpot's state relative to the sofa-table frame as "Middle Right". Furthermore, consider τ_2 to be another known QSR that describes the charger's state relative to the table-flowerpot frame as "Middle Left". Given τ_1 and τ_2, the robot can compose a new QSR, τ_3, which describes the charger's state relative to the sofa-table frame, as "Top Right". For illustration, see Fig. 1b, where the sofa, table, flowerpot, and

charger, correspond to landmarks A, B, C, and D. The robot then can use τ_3 to infer its target heading.

In this work, we introduce a general approach to incorporate compositions in qualitative frameworks, aiming to further advance the state-of-the-art. Before stating our contributions, we briefly review the most relevant work done in the field to trace the existing gaps.

A. Related Work

Qualitative Spatial Reasoning applications for various robotics tasks began to emerge about three decades ago.

Naturally, passive aspects were the first to be addressed. A pioneer work by [2] presented a novel approach for egocentric robot localization based on the relative ordering of observed landmarks. [3] and [4] further improved this idea by encoding ordering views in a more complex hence distinguishable fashion, enabling localization ambiguity reduction. Freksa suggested in [1] to represent the qualitative location of a landmark relative to a boundary line settled by a pair of other landmarks used as a reference. The location is described as "to the left" or "to the right" of the boundary. Freksa and Zimmermann further refined this binary partitioning of space in [5], into quadratic and hexagonal ones, by adding extra boundary lines perpendicularly crossing the original one. The latter mentioned partitioning forms known as the "Freksa's Single Cross" (FSC) and "Freksa's Double Cross" (FDC).

Moreover, in [1], Freksa introduced the binary composition operator that allows inference about the qualitative relationships between landmarks not directly observed together. [6] presented a set of qualitative spatial constraints between oriented straight line segments formed by pair of landmarks (dipoles) to describe the environment. [7] and [8] further extended this work and formulated a bipole-based composition operator.

McClelland et al. introduced a comprehensive method for qualitative autonomous localization and mapping in [9]. The proposed algorithm constructs a graph-based map that encodes the environment using the relative geometrical layout of landmark triplets. For each triplet, one landmark is estimated in a local frame defined by the other two. The landmark is associated with one of several possible qualitative states, considering the FDC partitioning. The authors extended their work in [10] by incorporating a method determining the qualitative states of landmarks based on a novel set of geometric constraints. The above yielded a new qualitative spatial partitioning called the "Extended Double Cross" (EDC). This partition and those mentioned above are illustrated in Fig. 2. In addition, this work contributed a new composition operator, suitable for triplets, formulated as a look-up table. Another followed-up paper generalized the latter by developing a probabilistic QRM method (PQRM, [11]). In [12], Mor and Indelman were the first to incorporate stochastic motion model constraints in their formulations, reducing uncertainty levels of both landmarks and robot trajectory estimations. In addition, the authors contributed a novel derivation of a probabilistic composition.

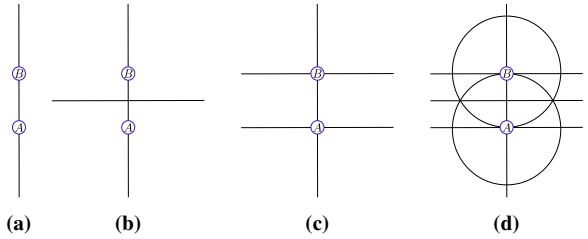

Figure 2: Four different partitions of the metric space. (a) binary left-right [1]; (b) Freksa Single Cross (FSC, [5]); (c) Freksa Double Cross (FDC, [5]); (d) Extended Double Cross (EDC, [10]); Consider the triplet $AB{:}C$ for example. The reference landmarks A and B fix the local frame (as illustrated), and the location of C is evaluated in terms of the disjoint regions defined by the partition, known as qualitative states.

B. Contributions

While the composition has been formulated in numerous ways in past works (see Section I-A), two fundamental assumptions have been made in all cases.

Firstly, the identity of the target QSR was assumed to be externally given. In practice, given a set of QSRs to start with, one would seek to know what new QSRs can be formed using compositions. Consider the example from Section I, given the initial set of τ_1, τ_2, and potentially more QSRs, τ_3 is only one possible target QSR among many others. Secondly, past works have assumed that the source QSRs needed to compose a target one are externally given and available. However, in general, they may be several alternatives to choose the source QSRs. Furthermore, at least one source might be unavailable and requires additional compositions to form it. This issue may occur recursively. Namely, there may be multiple composition sequences to form a target QSR, and it is unclear how to choose the optimal one. In the example, rather than composing τ_3 using τ_1 and τ_2, we could have chosen a different strategy if the initial QSRs set was richer.

These two issues are addressed in this paper.

II. Notations And Problem Formulation

We consider a robot operating in a 2D environment, consisting of a known set of landmarks, denoted by $\mathcal{L}$. The robot is entrusted with a valuable task, such as planning its next course of action to reach a desirable destination. It maintains a map consisting of Qualitative Spatial Relationships (QSRs) between triplets of landmarks. For each triplet, a target landmark is localized relative to a local landmark-centric frame based on the other two. A predefined partition divides the space into a finite set of qualitative states (Fig. 2), and the target landmark is associated with one of them, using methods such as in [11] and [12]. We stress that these methods evaluate the triplets while assuming data association is solved. Accordingly, we assume the same in this work.

Let $\mathcal{T}$ denote the set of triplets evaluated by the robot. In the following, we shall refer to $\mathcal{T}$ as the set of source triplets. We consider only triplets in this work, as this is the most basic and standard case, and since richer QSRs can always be broken down into ternary ones. We exclude binary QSRs, as these are relevant when dealing with complex volumed

landmarks (extended landmarks, see [13] and [14]), while we are assuming point landmarks in order to be aligned with the majority of works in this field.

As part of the robot's task, evaluating new triplets via composition may be helpful. Topologically speaking, to compose a target triplet, using a single composition operation, we need a pair of landmark triplets, such that their intersection size equals 2 (namely, they share two landmarks in common), and their union contains all three landmarks that form the target triplet (see [10]). The mutual landmarks allow us to fix both triplets relative to the same frame and hence to infer (compose) relationships between new combinations of triplets (consisting of landmarks taken from the union). For example, given the triplets $AB{:}C$ and $BC{:}D$, where the landmark after the colon is the target one, we can compose $AB{:}D$ (see Fig. 1b). Similarly, using $AC{:}B$ and $CB{:}D$, we can compose $AC{:}D$. Note that all ordered permutations of a given triplet are available using additional unary manipulations, via the operators "INVERSE", "LEFT", and "RIGHT" defined in [10]. Consequently, given the triplets ABC and BCD (and by omitting the ':' notation, we mean, no matter the permutation), we can compose either ABD or ACD (all possible permutations). Hence, from this point on, we shall omit the colon notation. The above topological rule, which we use for the rest of this paper, is formulated in the following Lemma:

Lemma 1. A triplet τ can be composed using a single composition operation (or directly) based on the triplets τ_1 and τ_2, if the following hold:

1) $|\tau_1 \cap \tau_2| = 2$
2) $\tau \subset \tau_1 \cup \tau_2$

Note that a triplet of landmarks, $\tau = L_1 L_2 L_3$, can be explicitly written as a set, i.e., $\{L_1, L_2, L_3\}$. The latter version is the preferable one in the above Lemma and in other formal parts that follow.

One difficulty that may arise when performing compositions stems from recursive aspects. Consider a target triplet to compose τ. The triplets τ_1 and τ_2 required to perform the composition according to Lemma 1 may not be available (i.e., not in $\mathcal{T}$), and therefore, we would have to compose them first. This issue may occur repeatedly. Consequently, to compose τ, we need to perform a concrete sequence of composition operations, relying on source triplets from $\mathcal{T}$, that creates τ as a final outcome.

We aim to address two main issues arising from the above. The first refers to the initial set of source triplets. Intuitively, given only a sparse set of sources $\mathcal{T}$, we can compose only some triplets in $\mathcal{L}$ or worse, none. For that reason, we first identify and formulate a sufficient topological condition required for $\mathcal{T}$ so that any desired triplet in $\mathcal{L}$ would be feasible to compose (Section III-A). The second issue is that there are often multiple alternatives for generating a sequence of composition operations to create a target triplet. We propose an algorithm that generates the optimal one in terms of a predefined cost (Section III-B).

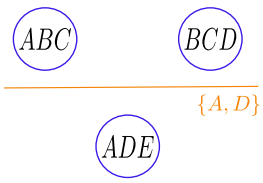

Figure 3: Demonstration of definitions 1-4. An example for a set of triplets, $\mathcal{T}=\{ABC, BCD, ADE\}$, is presented. The *Landmark Space* of $\mathcal{T}$ is $\mathcal{L}(\mathcal{T})=\{A, B, C, D, E\}$. The line represents the *Cut* $C=(\mathcal{T}_L=\{ABC, BCD\}, \mathcal{T}_R=\{ADE\})$ of $\mathcal{T}$. Both landmarks A and D are common to the *Landmark Spaces* $\mathcal{L}(\mathcal{T}_L)$ and $\mathcal{L}(\mathcal{T}_R)$. Thus, C is 2-*common*. Note that C is also 1-*common* and 0-*common* by definition (but not 3-*common*). Finally, $\mathcal{T}$ is a *Composable* set under $\mathcal{L}(\mathcal{T})$ (or any $\mathcal{L} \subseteq \mathcal{L}(\mathcal{T})$), since C is 2-*common* such that: (1) $\mathcal{T}_L$ is *Composable* under $\mathcal{L}(\mathcal{T}_L)$, since the only non-trivial *Cut* $C' = (\mathcal{T}'_L=\{ABC\}, \mathcal{T}'_R=\{BCD\})$ is 2-*common* and both $\mathcal{T}'_L$ and $\mathcal{T}'_R$ are *Composable* under $L(\mathcal{T}'_L)$ and $\mathcal{L}(\mathcal{T}'_R)$, respectively ($|\mathcal{T}'_L|=|\mathcal{T}'_R|=1$). (2) $\mathcal{T}_R$ is *Composable* under $\mathcal{L}(\mathcal{T}_R)$, since $|\mathcal{T}_R|=1$.

III. APPROACH

A. Composable Set of Triplets

In this section, we establish the term of a *Composable* set of triplets. We prove that we can compose any target triplet in the relevant landmark space given a source set $\mathcal{T}$ of this particular topological form.

As a preliminary step, we first define the following auxiliary terms:

Definition 1. Let $\mathcal{T}$ be a set of triplets. The *Landmark Space* of $\mathcal{T}$, denoted by $\mathcal{L}(\mathcal{T})$, is defined as:

$$\mathcal{L}(\mathcal{T}) = \bigcup_{\tau \in \mathcal{T}} \tau \tag{1}$$

Note that the *Landmark Space* of a single triplet set is the triplet itself: $\mathcal{L}(\{\tau\})=\tau$.

Definition 2. Let $\mathcal{T}$ be a set of triplets. A *Cut* $C=(\mathcal{T}_L, \mathcal{T}_R)$ of $\mathcal{T}$, is a partition of $\mathcal{T}$ into two disjoint subsets, $\mathcal{T}_L$ and $\mathcal{T}_R$, s.t. $\forall \tau \in \mathcal{T}$, either $\tau \in \mathcal{T}_L$ or $\tau \in \mathcal{T}_R$, but not both.

Definition 3. Let $\mathcal{T}$ be a set of triplets and let $\alpha \in \mathbb{N} \cup \{0\}$. A *Cut* $C=(\mathcal{T}_L, \mathcal{T}_R)$ of $\mathcal{T}$ is called α-*common* if $|\mathcal{L}(\mathcal{T}_L) \cap \mathcal{L}(\mathcal{T}_R)| \geq \alpha$.

We are now ready to define the term of a *Composable* set of triplets.

Definition 4. Let $\mathcal{T}$ be a set of triplets and let $\mathcal{L}$ be a *Landmark Space*. We say that $\mathcal{T}$ is *Composable* under $\mathcal{L}$, if $\mathcal{L} \subseteq \mathcal{L}(\mathcal{T})$, and one of the following holds:

1) $|\mathcal{T}|=1$.
2) $|\mathcal{T}|>1$ and **there is** a 2-*common Cut* $C=(\mathcal{T}_L, \mathcal{T}_R)$ of $\mathcal{T}$, s.t. $\mathcal{T}_L$ is *Composable* under $\mathcal{L}(\mathcal{T}_L)$ and $\mathcal{T}_R$ is *Composable* under $\mathcal{L}(\mathcal{T}_R)$.

Fig. 3 demonstrates the above definitions.

We now aim to prove that we can compose any triplet $\tau \subseteq \mathcal{L}$, given a *Composable* set under $\mathcal{L}$.

Theorem 1. Let $\mathcal{T}$ be a *Composable* set of triplets under the *Landmark Space* $\mathcal{L}$. Then any triplet $\tau \subseteq \mathcal{L}$ can be composed based on triplets from $\mathcal{T}$.

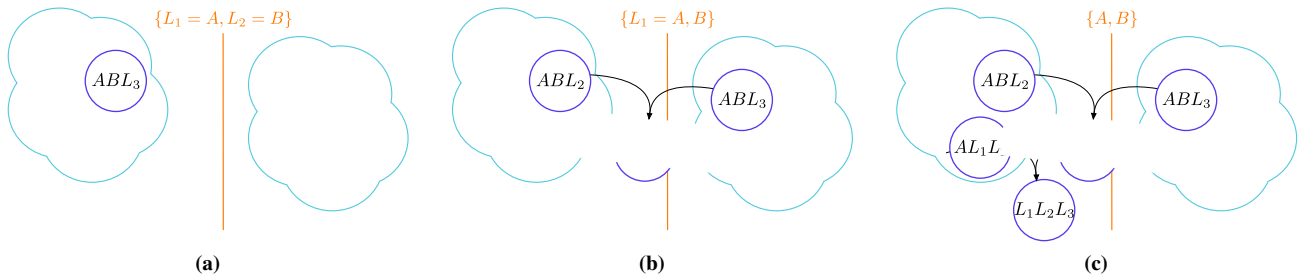

Figure 4: An illustration for the three cases described in Theorem 1's proof. (a) case 1; (b) case 2; (c) case 3; Triplets are shown inside circles. The line represents the *Cut* that splits the original set, $\mathcal{T}$, into two disjoint ones, $\mathcal{T}_L$ and $\mathcal{T}_R$ (represented by the clouds), with the {text} specifies landmarks that $\mathcal{L}(\mathcal{T}_L)$ and $\mathcal{L}(\mathcal{T}_R)$ share in common. Finally, a pair of triplets that points on a third one via black arrow represent that the latter is directly composed using the first two.

Proof. We prove Theorem 1 using induction on number of set elements (triplets), $|\mathcal{T}|$.

Base step: Suppose $|\mathcal{T}|=2$ $\big(|\mathcal{T}|=1$ is a trivial case$\big)$. $\mathcal{T}$ is *Composable* under $\mathcal{L}$, thus, the only non-trivial *Cut* exists in this case is 2-*common*. Without loss of generality (WLOG), suppose $\mathcal{T}=\{ABC,BCD\}$. Indeed, according to Lemma 1, we can compose ABD and ACD, i.e., all other triplets exist in $\mathcal{L}(\mathcal{T})$ (and thus also in $\mathcal{L}$, since $\mathcal{L}\subseteq\mathcal{L}(\mathcal{T})$).

Induction step: Suppose any triplet $\tau\subseteq\mathcal{L}$ can be composed based on triples from $\mathcal{T}$, for all $1\leq|\mathcal{T}|\leq n$. We prove that the same is true for $|\mathcal{T}|=n+1$.

Suppose $|\mathcal{T}|=n+1$ and let $\tau=L_1L_2L_3$ be a triplet in $\mathcal{L}$. We show that τ can be composed using triplets from $\mathcal{T}$. Since $\mathcal{T}$ is *Composable* under $\mathcal{L}$, we are guaranteed that it has a 2-*common Cut*, $(\mathcal{T}_L,\mathcal{T}_R)$, s.t. $\mathcal{T}_L$ is *Composable* under $\mathcal{L}(\mathcal{T}_L)$ and $\mathcal{T}_R$ is *Composable* under $\mathcal{L}(\mathcal{T}_R)$.

Suppose, WLOG, that $\{A,B\}\subseteq\mathcal{L}(\mathcal{T}_L)\cap\mathcal{L}(\mathcal{T}_R)$. We examine three possible cases (see illustration in Fig. 4).

Case 1 : $|\{L_1,L_2,L_3\}\cap\{A,B\}|=2$. WLOG, we assume that $L_1=A$ and $L_2=B$ and continue examining L_3. The latter must be in $\mathcal{L}(\mathcal{T}_L)$, or $\mathcal{L}(\mathcal{T}_R)$, or both. WLOG, suppose $L_3\in\mathcal{L}(\mathcal{T}_L)$. Thus, we are guaranteed that $\{A,B,L_3\}\subseteq\mathcal{L}(\mathcal{T}_L)$, and consequently ABL_3 (namely, $L_1L_2L_3$) can be composed according to the assumption since $\mathcal{T}_L$ is *Composable* under $\mathcal{L}(\mathcal{T}_L)$ and $|\mathcal{T}_L|\leq n$.

Case 2 : $|\{L_1,L_2,L_3\}\cap\{A,B\}|=1$. WLOG, we assume that $L_1=A$ and continue examining L_2,L_3. If they are both in $\mathcal{L}(\mathcal{T}_L)$ or both in $\mathcal{L}(\mathcal{T}_R)$, we finished (similarly to case 1). Otherwise, WLOG, we assume that L_2 is exclusively in $\mathcal{L}(\mathcal{T}_L)$ and L_3 is exclusively in $\mathcal{L}(\mathcal{T}_R)$. According to the assumption, we are guaranteed that ABL_2 and ABL_3 can be composed based on $\mathcal{T}_L$ and $\mathcal{T}_R$, respectively. Finally, using these two triplets, we can compose AL_2L_3 (Lemma 1), namely, $L_1L_2L_3$.

Case 3 : $|\{L_1,L_2,L_3\}\cap\{A,B\}|=0$. If $\{L_1,L_2,L_3\}$ are all in $\mathcal{L}(\mathcal{T}_L)$ or all in $\mathcal{L}(\mathcal{T}_R)$, we finished (similarly to case 1). Otherwise, WLOG, we assume that L_1 and L_2 are exclusively in $\mathcal{L}(\mathcal{T}_L)$ and L_3 is exclusively in $\mathcal{L}(\mathcal{T}_R)$. According to the assumption, we are guaranteed that ABL_2 and AL_1L_2 can be composed based on $\mathcal{T}_L$, and that ABL_3 can be composed based on $\mathcal{T}_R$. Using ABL_2 and ABL_3, we can compose AL_2L_3 (Lemma 1). Finally, using AL_1L_2 and AL_2L_3, we can compose $L_1L_2L_3$ (Lemma 1)■.

B. Optimal Composition Sequence

As we saw in the previous section, given a *Composable* set of source triplets under the *Landmark Space* $\mathcal{L}$, we can compose any triplet in $\mathcal{L}$. Once we chose a target triplet to compose, we aim to find an appropriate sequence of composition operations to create it, considering the recursive aspects discussed in Section II. Topologically, such a sequence can be described as a binary tree representing all compositions required to create a target triplet. Each tree node represents a triplet to compose, with its child nodes representing the triplets pair required to composed it (directly). The root specifies the target triplet, while the leaves specify source triplets taken from $\mathcal{T}$ (the initial set of source triplets which considered given). We name this binary tree a *Composition-Tree*. Its formal definition is given below:

Definition 5. A *Composition-Tree* is a binary tree, $T=(V,E)$, where:

1) Each node $v_\tau\in V$ represents a triplet of landmarks τ.
2) Each node $v_\tau\in V$ has either two children, $v_{\tau_L},v_{\tau_R}\in V$, representing that τ is composed directly using τ_L and τ_R, or none, if τ is a source triplet (in which case, v_τ is a leaf).

See Figs. 5e-5f for illustration. In general, many topologically suitable trees may exist for a target triplet. So it is still a question, how to select the best one among all possibilities.

Formally, consider the set of source triplets $\mathcal{T}$, and a target triplet to compose, τ_o. We denote by $\mathbb{T}_{\tau_o}$ the set of all possible *Composition-Tree*s with τ_o being the root and leaves taken from $\mathcal{T}$. We aim to find the optimal *Composition-Tree*, considering all instances from $\mathbb{T}_{\tau_o}$, in terms of a predefined cost C (see cost specifications in Section III-B2). That is:

$$T^*=\underset{T\in\mathbb{T}_{\tau_o}}{\arg\min}\sum_{\tau\in T}C(\tau). \tag{2}$$

We propose a three-stage algorithm for solving the above problem, given an initial set of source triplets, $\mathcal{T}$. First, we initialize a unique topological structure called a *Composition-Graph* (Section III-B1). The graph sets the infrastructure for the cost maintenance regarding the different triplets that exist

in $\mathcal{L}(\mathcal{T})$. Next, we show how to update the graph every time a new source triplet is being considered (Section III-B2). Finally, given a query triplet to compose, $\tau_o \subseteq \mathcal{L}(\mathcal{T})$, we extract from the graph the *Composition-Tree* representing the optimal sequence of compositions to form τ_o. In case no valid sequence exists, the *Composition-Tree* will be empty (Section III-B3).

Alg. 1 summarizes the above. Further aspects regarding the algorithm are analyzed in Section III-B4. A comprehensive running example can be found in Section III-C.

Algorithm 1: Optimal Composition Sequence

Input: Set of source triplets $\mathcal{T}$, target triplet $\tau_o \subseteq \mathcal{L}(\mathcal{T})$

▷ Initialize empty graph
$G \leftarrow$ InitCompGraph$\big(\mathcal{L}(\mathcal{T})\big)$
▷ Update Graph
for $\tau \in \mathcal{T}$ **do**
  $G \leftarrow$ UpdateCompGraph(G,τ)
▷ Extract τ_o optimal *Composition-Tree*
$T \leftarrow$ ExtractCompTree(G,τ_o)
return T

1) Composition-Graph Initialization:

The *Composition-Graph* is a useful topological representation that allows propagating costs between triplets easily. Its exact structure is defined as follows:

Definition 6. A *Composition-Graph* is a bipartite graph, $G=(V_\tau,V_q,E)$ where:

1) Each node $v_\tau \in V_\tau$ represents a triplet of landmarks τ.
2) Each node $v_q \in V_q$ represents a sub-space of four landmarks (a quartet) q.
3) There is an edge $e=(v_\tau,v_q) \in E$, where $v_\tau \in V_\tau$ and $v_q \in V_q$, if and only if $\tau \subset q$.

The above representation is motivated by Lemma 1, where each quartet node connects exactly four triplets nodes with a direct composition relationship. Meaning, each pair of triplets taken from this foursome can be used to compose one of the remaining two, using a single composition. Alg. 2 shows how to construct and initialize a new *Composition-Graph*, considering a given *Landmark Space* $\mathcal{L}$. An illustration for the *Composition-Graph* initialization can be found in Fig. 5a. Note that we initialize a complete *Composition-Graph*, that is, it contains all possible $\binom{|\mathcal{L}|}{3}$ triplet and $\binom{|\mathcal{L}|}{4}$ quadrants that exist in $\mathcal{L}$, since potentially we aim to use it to compose any triplet in $\mathcal{L}$. Furthermore, each triplet node v_τ is initialized with a cumulative cost value (denoted by d) of ∞, which is later updated considering the arrival of new information regarding source triplets. The cumulative cost represents the total cost of the current optimal *Composition-Tree* of τ embedded in the graph. In addition, each v_τ is initialized with an empty parents list, later updated with the nodes representing the triplets designated to compose τ directly. Note that the parents (in terms of the *Composition-Graph*) are later referred to as children (in terms of the extracted *Composition-Tree*).

Algorithm 2: Initialize Composition Graph

Function InitCompGraph(*Landmark space* $\mathcal{L}$)**:**
  ▷ Initialize empty graph
  $G \leftarrow \{V_\tau \leftarrow \emptyset, V_\mathcal{Q} \leftarrow \emptyset, E \leftarrow \emptyset\}$
  ▷ Initialize triplets nodes
  $\mathcal{T}_\mathcal{L} \leftarrow$ Set of all $\binom{|\mathcal{L}|}{3}$ triplets residing in $\mathcal{L}$
  for $\tau \in \mathcal{T}_\mathcal{L}$ **do**
    Add node v_τ to V_τ representing τ
    $d(v_\tau) \leftarrow \infty$ ▷ Cumulative Cost
    Parents$(v_\tau) \leftarrow \emptyset$
  ▷ Initialize quartets nodes and edges
  $\mathcal{Q}_\mathcal{L} \leftarrow$ Set of all $\binom{|\mathcal{L}|}{4}$ quartets residing in $\mathcal{L}$
  for $q \in \mathcal{Q}_\mathcal{L}$ **do**
    Add node v_q to $V_\mathcal{Q}$
    $\{v_{\tau_1},v_{\tau_2},v_{\tau_3},v_{\tau_4}\} \leftarrow$ nodes from V_τ representing triplets residing in q
    for $v \in \{v_{\tau_1},v_{\tau_2},v_{\tau_3},v_{\tau_4}\}$ **do**
      Add edge (v_q,v) to E
  return G

2) Composition-Graph Update:

Given a source triplet $\tau \in \mathcal{T}$, we aim to properly update our *Composition-Graph* by updating the cumulative cost of v_τ and all affected triplets nodes. To that end, we define the following cost:

$$C(\tilde{\tau}) = \begin{cases} C^{\text{source}}(\tilde{\tau}), & \text{if } \tilde{\tau} \text{ is a source triplet} \\ C^{\text{comp}}(\tilde{\tau}_L,\tilde{\tau}_R), & \text{if } \tilde{\tau} \text{ is composed directly using } \tilde{\tau}_L,\tilde{\tau}_R \end{cases}$$

, where C^{source} can be any real function and where C^{comp} can be any symmetric non-negative real function. The cost is desined by the user to consider a desired criterion of interest. A specific selection example is given in Section III-C.

We shall now explain in detail how we update the *Composition-Graph*, considering the arrival of new information regarding the source triplet τ.

First, we calculate $c=C^{\text{source}}(\tau)$, the new candidate cumulative cost of τ resulted from the newly arrived information. We then locate the node representing τ, v_τ, and check whether c is better (lower) than the cumulative cost currently assigned to it, $d(v_\tau)$. If so, we update v_τ with the new cost c and reset its parent list, stating that τ is qualified to be used as a source triplet.

In case of an update, we continue examining all quadrant nodes adjacent to v_τ. For each adjacent node v_q, we look at the other three triplets nodes connected to it. Let v_{τ_1},v_{τ_2}, and v_{τ_3} denote these nodes. We check for each one if we can improve its cumulative cost as a result of the recently updated one of v_τ. Consider v_{τ_1} for example, we compare its current cumulative cost, $d(v_{\tau_1})$, with the candidate ones $c_i=d(v_\tau)+d(v_{\tau_i})+C^{\text{comp}}(\tau,\tau_i), \forall i \in \{2,3\}$. If for some $i \in \{2,3\}$ c_i improves $d(v_{\tau_1})$, we assign it to v_{τ_1} and update

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/LRA.2022.3144525, IEEE Robotics and Automation Letters

its parents to be v_τ and v_{τ_i}, stating that τ_1 is designated to be composed directly using v_τ and v_{τ_i}. For each newly updated triplet node, we go through the same process until there are no more updates.

The *Composition-Graph* is updated incrementally, each time based on a single source triplet from $\mathcal{T}$. Note that since the update rule obeys Lemma 1 topologically, Theorem 1 determines which nodes are expected to be updated based on the topological structure of $\mathcal{T}$ alone. That is, if $\mathcal{T}$ is *Composable* under the *Landmark Space* $\mathcal{L}$, then each node representing a triplet τ in $\mathcal{L}$ is guaranteed to be updated with a finite cost value. The latter means that we can extract a non-empty *Composition-Tree* describing the optimal sequence of composition operations required to compose τ. Of course, even when all the triplets nodes are assigned with finite costs, extra updates can only lead to further improvement.

Alg. 3 formalizes the above.

Algorithm 3: Composition Graph Update

Function UPDATECOMPGRAPH (*Composition Graph* G*, triplet to update* τ)**:**
- $\mathcal{Q} \leftarrow \emptyset$ ▷ Initialize empty set
- $v_\tau \leftarrow$ node from G representing τ
- ▷ Update v_τ
- $c{=}C^{\text{source}}(\tau)$
- **if** $c{<}d(v_\tau)$ **then**
  - $d(v_\tau) \leftarrow c$
  - $Parents(v_\tau) \leftarrow \emptyset$
  - $\mathcal{Q} \leftarrow \mathcal{Q} \cup v_\tau$
- ▷ Update graph (propagate)
- **while** $\mathcal{Q}$ *not empty* **do**
  - $v \leftarrow$ select and remove $v \in \arg\min_{v \in Q} d(v)$
  - **for** $v_q \in Adj(v)$ **do**
    - $V_\tau^{\text{Updated}} \leftarrow$ UPDATESTEP(G, v_q, v)
    - $\mathcal{Q} \leftarrow \mathcal{Q} \cup V_\tau^{\text{Updated}}$
- **return** G

Function UPDATESTEP (*Composition Graph* G*, Quartet node* v_q*, updated triplet node* v_τ)**:**
- $V_\tau^{\text{Updated}} \leftarrow \emptyset$
- $V_\tau^{\text{adj}} \leftarrow Adj(v_q) \backslash \{v_\tau\}$
- ▷ Propagate cost in case of improvement
- **for** $v \in V_\tau^{adj}$ **do**
  - **for** $\tilde{v} \in V_\tau^{adj} \backslash \{v\}$ **do**
    - $v_{\tau_L} \leftarrow v_\tau, v_{\tau_R} \leftarrow \tilde{v}$
    - $c \leftarrow d(v_{\tau_L}) + d(v_{\tau_R}) + C^{\text{Comp}}(\tau_L, \tau_R)$
    - **if** $c{<}d(v)$ **then**
      - $d(v) \leftarrow c$
      - $Parents(v) \leftarrow \{v_{\tau_L}, v_{\tau_R}\}$
      - $V_\tau^{\text{Updated}} \leftarrow V_\tau^{\text{Updated}} \cup \{v\}$
- **return** V_τ^{Updated}

3) Composition-Tree Extraction:

Given an updated *Composition-Graph* and a finite-cost node v_{τ_o}, we aim to extract the optimal *Composition-Tree* of the corresponding triplet τ_o (Eq. 2).

Alg. 3 assigns to each node its optimal parents in terms of yielded cumulative cost. The latter converted to be the node's children in the embedded *Composition-Tree*. Accordingly, all we have left is to recursively extract the *Composition-Tree* embedded in the graph, starting from the root v_{τ_o}. We stop when we reach nodes representing source triplets, which are reflected as the tree leaves. Alg. 4 sums the recursive extraction process.

Algorithm 4: Extract *Composition-Tree*

Function EXTRACTCOMPTREE (*Composition Graph* G*, target triplet* τ_o)**:**
- Initialize empty tree T
- ▷ Stopping Criterion
- $v_{\tau_o} \leftarrow$ node from G representing τ_o
- **if** $d(v_{\tau_o}){=}\infty$ **then**
  - **return** T
- ▷ Construct tree
- $Root(T) \leftarrow v_{\tau_o}$
- $\{v_{\tau_L}, v_{\tau_R}\} \leftarrow Parents(v_{\tau_o})$
- $Left(T) \leftarrow$ EXTRACTCOMPTREE(G, τ_L)
- $Right(T) \leftarrow$ EXTRACTCOMPTREE(G, τ_R)
- **return** T

4) Algorithm 1 Analysis:

In this section, we analyze the correctness of Alg. 1 and explain why its convergence is guaranteed. In addition, we provide complexity analysis regarding time and place.

To analyze our algorithm, we shall first identify that the update step (Alg. 3) is an instance of a generalized version of *Dijkstra* [15] for directed *Hyper-Graph*s, called *SBT-Dijkstra* [16]. Recall that a directed *Hyper-Graph* is a pair $G{=}(V,E)$, where $V{=}\{v_1, v_2, \ldots, v_n\}$ is the set of nodes and $E{=}\{e_1, e_2, \ldots, e_m\}$, with , $e_i{=}\big(T(e_i){\subseteq}V, H(e_i){\subseteq}V\big)$, $\forall i{=}1,2,\ldots,m$, is the set of directed *hyperedge*s. T and H known as the *tail* and *head* of the *hyperedge*. Note that the above definition generalizes the standard directed graph (if $|T(e_i)|{=}|H(e_i)|{=}1, \forall i{=}1,2,\ldots,m$, we get a standard directed graph). Alternatively, we could have defined the *Composition-Graph* as a directed *Hyper-Graph*, where each directed *hyperedge* connects a foursome of triplets having a direct composition relationship. Specifically, two out of the four triplets form the *tail*, whereas the remaining two form the *head*, stating that each *head* triplet can be composed directly using the pair of *tail* ones. Consequently, we would replace each quartet node in the original *Composition-Graph* with a set of 6 directed *hyperedge*s, since we have $\binom{4}{2}{=}6$ ways of choosing pair of tail triplets (the head is dictated given that choice). Moreover, accordingly, the cost C^{comp} would now be the weight of the *hyperedge*s.

Correctness. The correctness of each update step is guaranteed based on the correctness of the *SBT-Dijkstra* procedure. Similarly to the original *Dijkstra* algorithm, it can be

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/LRA.2022.3144525, IEEE Robotics and Automation Letters

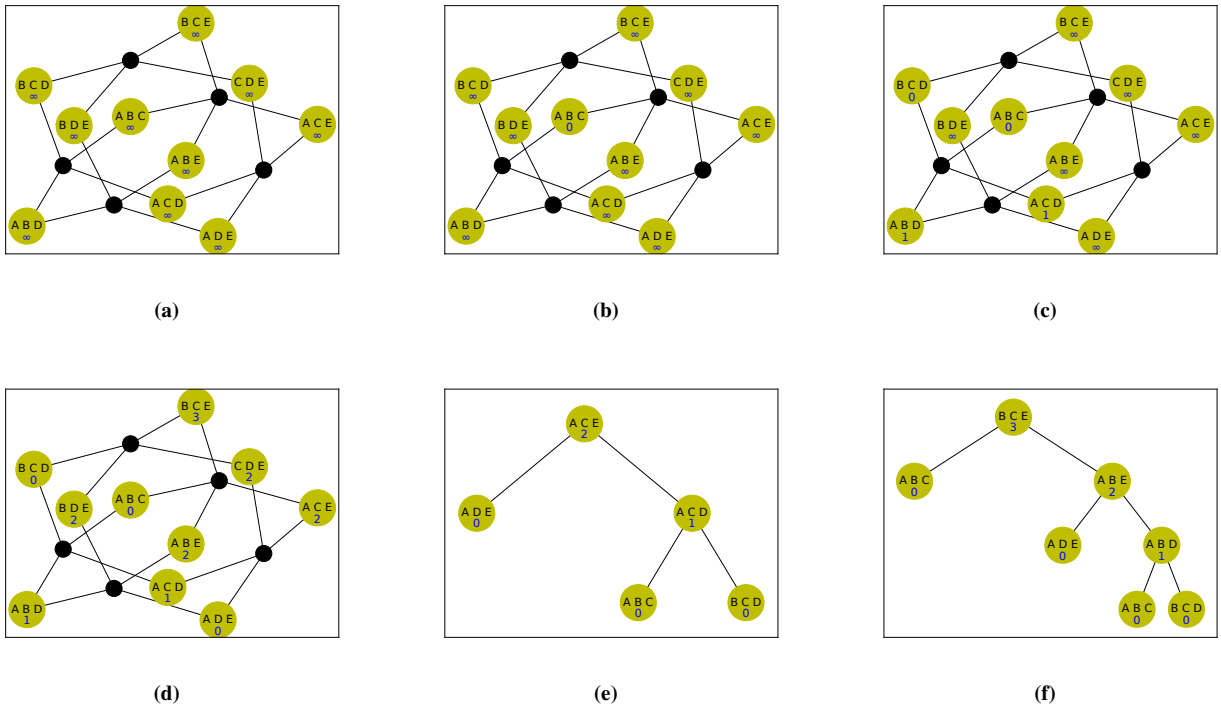

Figure 5: A running example of Alg. 1, as described in Section III-C. (a) illustrates *Composition-Graph* initialization, as an output of Alg. 2, with $\mathcal{L}=\{A,B,C,D,E\}$. Triplets nodes are shown in mustard color, whereas quartet nodes are in black; (b)-(d) demonstrate the graph update (Alg. 3), considering the cost described in Section III-C. The outcome graphs of three consecutive update steps, in which the cost of ABC, BCD, and ADE is set to 0, are shown. Since the set of triplets mentioned above is *Composable* under $\mathcal{L}$, the update of the entire graph is guaranteed. Note that after the second step (c), all four triplets exist under the sub-space $\{A,B,C,D\}$ are updated, as $\{ABC,BCD\}$ is a *Composable* set under that space. Moreover, the same is true if we considering the first step (b) alone and the *Landmark Space* $\{A,B,C\}$ (however, the latter is a degenerate case, as no extra triplets exist in this sub-space); (e)-(f) Two *Composition-Trees* extracted from the *Composition-Graph* (Alg. 4);

proven using induction over the number of visited nodes (i.e., nodes pulled out from $\mathcal{Q}$). As explained in [16], a crucial key point to be ensured is that no negative cycle is detected during the algorithm's operations, as it might lead to cyclic costs improvements of nodes in the graph. Indeed, in our case, all cycles are nondecreasing, as node cumulative costs are being aggregated additively and since C^{comp} is nonnegative. While the original *Dijkstra* detects shortest paths, the *SBT-Dijkstra* detects shortest *hyperpath*s (i.e., sequences of nodes and *hyperedge*s). In our case, the *hyperpath* is equivalent to an embedded *Composition-Tree* branch. It is important to note that applying the update step multiple times, as we do in Alg. 1, does not affect its correctness (the same induction proof mentioned above still holds). As long as we keep the cumulative costs from previous update steps, Alg. 1 is guaranteed to terminate with the optimal embedded branches, considering all source triples (multiple sources shortest paths). For that reason, we only initialize the cumulative cost of each triplet node once, during Alg. 2. Consequently, at the end of all update steps, any finite cost node can be reversely opened into the shortest *Composition-Tree* rooted at that node.

Considering the above, and given a target triplet to compose, τ_o, we are guaranteed that Alg. 4 extracts the shortest *Composition-Tree* originating in v_{τ_o}, i.e., $T^*=\arg\min_{T\in\mathbb{T}_{\tau_o}} d(v_{\tau_o})$, where $\mathbb{T}_{\tau_o}$ is the set of all possible *Composition-Tree*s with v_{τ_o} as root. Assuming v_{τ_L} and v_{τ_R} are the parents (children in terms of the embedded *Composition-Tree*) assigned to v_{τ_o}, we can rewrite the above as $T^*=\arg\min_{T\in\mathbb{T}_{\tau_o}} d(v_{\tau_L})+d(v_{\tau_R})+C(v_{\tau_o})$. We can repeat this recursively until we reach nodes representing source triplets, which have no parents (*Composition-Tree* leaves). We get $T^*=\arg\min_{T\in\mathbb{T}_{\tau_o}} \sum_{\tau\in T} C(\tau)$ (Eq. 2).

Complexity. Time complexity: *Composition-Graph* initialization is $\mathcal{O}(|V_\tau|+|V_q|+|E|)$. Then, each update step is $\mathcal{O}(\max\{|V_\tau|+|V_q|+|E|,|V_\tau|^2\})$, as the total cost of node selection and removals from Q is $\mathcal{O}(|V_\tau|^2)$ and the total cost of propagating costs in the graph is $\mathcal{O}(|V_\tau|+|V_q|+|E|)$ (based on [16]). Assuming a set of k source triplets to update the *Composition-Graph* with, we get $\mathcal{O}\big(\max\{|V_\tau|+|V_q|+|E|,|V_\tau|^2\}\cdot k\big)$. The extraction stage is $\mathcal{O}(|V_\tau|)$ at most, in case the target *Composition-Tree* consists of all the nodes in V_τ. In total, we have $\mathcal{O}\big(\max\{|V_\tau|+|V_q|+|E|,|V_\tau|^2\}\cdot k\big)$.

Space complexity: here, the initialization is the bottleneck. We have $\mathcal{O}(|V_\tau|+|V_q|+|E|)$.

C. Running Example

We shall now demonstrate Alg. 1 via a running example. Consider the initial set of source triplets, $\mathcal{T}=\{ABC,BCD,ADE\}$, and suppose we aim to compose the triplets ACE and BCE, using minimum composition

operations, based on triplets from $\mathcal{T}$. Note that Theorem 1 guarantees that the above can be done since $\mathcal{T}$ is composable under the *Landmark Space* $\mathcal{L}$=$\{A,B,C,D,E\}$, which contains both target triplets. We show how we infer the optimal (i.e., minimal) sequence of composition operations to create these triplets.

First, we formulate the following cost function:

$$C(\tau)=\begin{cases}0, & \text{if } \tau \text{ is a source triplet}\\ 1, & \text{if } \tau \text{ is composed directly using } \tau_L,\tau_R,\end{cases}$$

i.e., a triplet τ is assigned with the cost value of 0 if it is a source triplet (since we need 0 composition operations to form it), and with 1 if it is designated to be composed directly using the pair τ_L and τ_R (since we need a single composition to create τ, given τ_L and τ_R).

We start by initializing an appropriate *Composition-Graph* (Alg. 2), with $\mathcal{L}$ as the input *Landmark Space* (Fig. 5a). We then perform the update step (Alg. 3) for each source triplet in $\mathcal{T}$. The outcome graphs after update steps for ABC, BCD and ADE are illustrated in Fig. 5b-5d, respectively. At the end of these three update steps, every triplet node in the graph is updated with a finite cumulative cost, which indicates (due to the particular cost function we chose) the minimal amount of composition operations required to form it. As Fig. 5d shows, two compositions are required to create ACE and three to create BCE.

Finally, we extract the optimal (minimal) sequence of compositions, each represented via a *Composition-Tree*, to create these target triplets (Alg. 4). Fig. 5e and Fig. 5f show the extracted trees for the target triplets. Recall that while Alg. 4 extracts the optimal *Composition-Tree* for a query triplet, many other non-optimal ones exist. For instance, BCE can also be composed directly using BDE and CDE, but such a choice would require at least five composition operations in total since both BDE and CDE require two composition operations each in the optimal case.

IV. CONCLUSIONS

In this paper, we addressed two main issues that arise regarding compositions calculi. First, given a set of prior qualitative spatial relationships between triplets of landmarks (source triplets), we have formulated a sufficient topological condition (*Composability*) attributed to the set, whose existence ensures the ability to compose an entire space of spatial relationships between new triplets. Secondly, we addressed the question of how these triplets should be composed. We presented a novel algorithm that finds the optimal way to compose a target triplet under an optimality criterion defined by the user. To the best of our knowledge, this algorithm is the first of its kind.

We encourage incorporating our algorithm as a component in future qualitative approaches for a variety of tasks, such as localization, mapping, and active planning. In planning tasks, one can formulate an objective function designed to turn a prior map (i.e., the set of qualitative relationships that form it) into a *Composable* one, which would allow it to be further expanded through composition. Future research works may also generalize our work to include qualitative relationships between pairs of landmarks as well (relaxing the assumption of point landmarks).

REFERENCES

[1] C. Freksa, "Using orientation information for qualitative spatial reasoning," in *Theories and methods of spatio-temporal reasoning in geographic space*. Springer, 1992, pp. 162–178.
[2] T. Levitt and D. Lawton, "Qualitative navigation for mobile robots," *Artificial Intelligence*, vol. 44, no. 3, 1990.
[3] C. Schlieder, "Representing visible locations for qualitative navigation," in *Qualitative Reasoning and Decision Technologies*. CIMNE, 1993, pp. 523–532.
[4] T. Wagner, U. Visser, and O. Herzog, "Egocentric qualitative spatial knowledge representation for physical robots," *Robotics and Autonomous Systems*, vol. 49, no. 1, pp. 25–42, 2004.
[5] C. Freksa and K. Zimmermann, "On the utilization of spatial structures for cognitively plausible and efficient reasoning," in *IEEE International Conference on Systems, Man and Cybernetics*, 1992, pp. 261–266.
[6] C. Schlieder, "Reasoning about ordering," in *Spatial Information Theory A Theoretical Basis for GIS*, A. U. Frank and W. Kuhn, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 1995, pp. 341–349.
[7] R. Moratz, L. Dominik, and M. Till, "A condensed semantics for qualitative spatial reasoning about oriented straight line segments," 2011.
[8] T. Mossakowski and R. Moratz, "Qualitative reasoning about relative direction of oriented points," *Artificial Intelligence*, vol. 180-181, pp. 34–45, 2012. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0004370211001251
[9] M. McClelland, M. Campbell, and T. Estlin, "Qualitative relational mapping for mobile robots with minimal sensing," *Journal of Aerospace Information Systems*, vol. 11, no. 8, pp. 497–511, 2014.
[10] ——, "Qualitative relational mapping and navigation for planetary rovers," *Robotics and Autonomous Systems*, vol. 83, pp. 73–86, 2016.
[11] J. Padgett and M. Campbell, "Probabilistic qualitative mapping for robots," *Robotics and Autonomous Systems*, vol. 98, pp. 292–306, 2017.
[12] R. Mor and V. Indelman, "Probabilistic qualitative localization and mapping," in *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2020.
[13] P. Fogliaroni, J. O. Wallgrün, E. Clementini, F. Tarquini, and D. Wolter, "A qualitative approach to localization and navigation based on visibility information," in *International Conference on Spatial Information Theory*. Springer, 2009, pp. 312–329.
[14] R. K. Goyal, "Similarity assessment for cardinal directions between extended spatial objects," Ph.D. dissertation, in Spatial Information Science and Engineering, University of Maine, 2000.
[15] E. Dijkstra, "A note on two problems in connexion with graphs," *Numerische Mathematik*, vol. 1, pp. 269–271, 1959.
[16] G. Gallo, G. Longo, S. Pallottino, and S. Nguyen, "Directed hypergraphs and applications," *Discrete applied mathematics*, vol. 42, no. 2-3, pp. 177–201, 1993.