Reasoning about Relative Position and Orientation of Moving Objects Using Answer Set Programming

Yusuf Izmirlioglu

August 20, 2024

Reasoning about Relative Position and Orientation of Moving Objects using Answer Set Programming

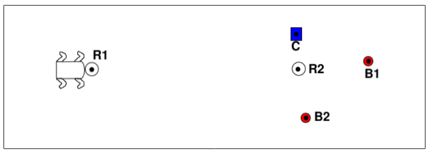
Yusuf Izmirlioglu

January 17, 2024

1 Introduction

Reasoning about position and distance is vital for cognitive robotics, land/marine navigation and surveying applications. Motion planning and collecting survey knowledge requires identifying relative position and orientation of static objects as well as navigating agents in the environment. In marine navigation, vessels need to position themselves with respect to other agents and the port, for example Vessel 4 reports vessel 2 is on its starboard 147^o. Vessel 3 states 6 is at his behind and near to the port, vessel 6 observes 7 on clock angle 9. As another example, in the robotic perception and motion planning problem of [Moratz *et al.*2005, Dylla and Moratz2004, Moratz and Ragni2008], the human user instructs the robot "to navigate to the red cube behind the blue cube" (Figure 1). To achieve this goal, the robot needs to identify which red cube is behind the blue cube (C). In a similar vein, oriented objects and relative position are also prevalent in surveying, localization/mapping and street networks, as in the statements "The cafeteria is in front-left of the library and very near to it", "When you go down the Kingston road and reach the junction, Wimbledon avenue will be on your left".

Figure 1: Robotic sensing and motion planning scenario



In these contexts, moving agents have some bearing and pivot objects may have some intrinsic orientation. Navigating robots and vessels need to know relative position of other agents for collision avoidance, motion/task planning and fleet management. Such relative location and distance information can be quantitative (e.g. the tanker robot is to the 12 m and 53 angle of robot 2) as well as qualitative or coarse (e.g. robot 5 is to the left-front and near to robot 7, ship 3 is starboard to ship 4). The latter is relevant in contexts with human presence or data obtained from sensors is imprecise or exact location of an extended object is hard to determine. In marine exploration, metric data about location and heading can be obtained from GPS and measurement devices, but additional complex calculations are necessary to determine relative location of the agent with respect to the port and other agents. In these sitautions, marine men often use simple view or deck sight to locate themselves, e.g. we are near and southwest of Patara marina, or the yatch is starboard and behind us. Besides such qualitative direct information is beneficial for verification of sensor data in case of fault or calibration. Until now, robotic and marine navigation literature mostly focused on collision and motion planning [Maaref and Barret2002, Frommberger2008, Miguel-Tomé2021], rather than reasoning about kowledge.

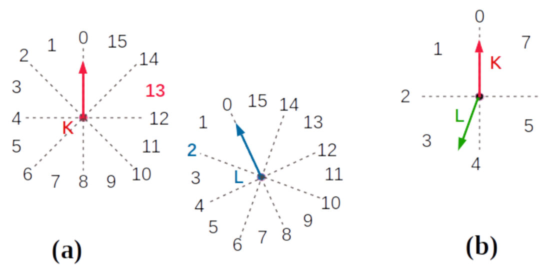
In this paper, we study reasoning about relative position, orientation and distance of point objects in 2D space. We start with Oriented Point Relation Algebra ($OPRA$)[Moratz *et al.*2005] which describes qualitative position of oriented points on the plane. We construct a new calculus $HOPRA$ by augmenting qualitative direction and quantitative location, distance, orientation information to $OPRA$. Then we develop a framework for reasoning with this hybrid formalism using Answer Set Programming. The framework can check consistency of a set of qualitative and quantitative constraints, explain source of inconsistency, handle uncertain and presumed information, infer new knowledge and generate a configuration of objects. We evaluate efficiency of our method by computational experiments and illustrate its applications with sample scenarios.

2 Preliminaries

2.1 OPRA Formalism

$OPRA$ describes qualitative position of two oriented points on 2D space with adjustable granularity g. In $OPRA_g$, there are $4g$ relations indexed from 0 to $4g-1$. The even indexed relations are linear (along a line) whereas odd indexed relations are angular. The length of an angular sector is $180^o/g$. Let (x_K, y_K) denote the location and ϕ_K denote the intrinsic orientation (heading) of object K. $\varphi_{KL} = tan^{-1}(y_L - y_K)/(x_L - x_K)$ denotes the absolute angle of the vector $\overrightarrow{KL}$ directed from K to L. If the location of K and L are different, the $OPRA_g$ relation is shown as $K\ {}_g\angle^i_j\ L$ which reads K is on sector i wrt viewpoint of L and L is on sector j wrt viewpoint of K. Namely, K falls onto sector i of L and L falls onto sector j of K. $\psi_{KL} = \varphi_{KL} - \phi_K$ denotes the relative angular position of L wrt K (and vice versa for ψ_{LK}). We name these relations *differential* OPRA relations. For example, in Figure 2(a) below $K\ {}_4\angle^2_{13}\ L$.

Figure 2: Example for OPRA relations



When the location of K and L coincide, the relative angular position of L with respect to K is defined as the difference between intrinsic orientations, i.e. $\delta_{LK} = \phi_L - \phi_K$. In this case, the $OPRA_g$ relation is shown as $K\ {}_g\angle\, i\ L$ which reads L is on sector i wrt viewpoint of K. We name this type of relation as *same* OPRA relation. To examplify, $K\ {}_2\angle\, 3\ L$ in Figure 2(b). Note that specifying j is unnecessary since $j = 4g - i$.

[Clementini *et al.*1997] has introduced qualitative interval-based distance relation with symbolic binary relation such as *overlap*, *near*, *far* with adjustable granularity. Each distance relation is associated with an interval, for example *near* corresponds to $[2.5,\ 4.7]$. If the Euclidean distance between two objects is in this range, they are *near* to each other. We create a new hybrid calculus named $HOPRA$ by adding distance relation $K\ \beta\ L$ and quantitative constraints to $OPRA$. Quantitative constraints are of the form
(1) $\phi(K) = a^o$, (2) $\phi(K) \in [a^o,\ b^o]$, (3) $\phi(K) = \overrightarrow{PR}$,
(4) $\psi(K, L) = a^o$, (5) $\psi(K, L) \in [a^o,\ b^o]$,

(6) $\delta(K,L) = a^o$, (7) $\delta(K,L) \in [a^o,\, b^o]$,
(8) *loc(K)* $= (x,y)$, (9) *dist(K,L)* $= c$, (10) *dist(K,L)* $\in [c,\, d]$.

(1-3) are about intrinsic orientation (heading), (4-5) about relative position for objects whose location different, (6-7) about relative position for objects with same location, (8) numerical location and (9-10) numerical distance.

In $HOPRA$, we also introduce *single-sided* differential OPRA relation $K\ {}_g\measuredangle^i\ L$. A single-sided OPRA relation specifies only the relative position of K wrt L, and not the other way around. The intuition is agent L may not know how it is observed from viewpoint of other agent K. In $HOPRA$, we allow for heterogenous relations: an agent K observes relative position of L with granularity g_1, while agent L observes K or M with granularity $g_2 \neq g_1$.

2.2 Answer Set Programming

Answer Set Programming (ASP) is a logic programming paradigm based on answer set semantics [Gelfond and Lifschitz19
Gelfond and Lifschitz1991]. It provides a formal framework for knowledge reasoning and declaratively solving computationally hard problems. ASP models a problem by a set of logical formulas (called rules), so that its models (called answer sets) characterize the solutions. ASP provides logical formulas, called rules, of the form

$$\textit{Head} \leftarrow L_1, \ldots, L_k, \textit{not } L_{k+1}, \ldots, \textit{not } L_l \tag{1}$$

where $l \geq k \geq 0$, *Head* is a literal and each L_i is a literal. A literal is an atom A or its negation $\neg A$) or $\bot$. A rule is called a *constraint* if *Head* is $\bot$, and a *fact* if $l = 0$. A set of rules is called a *program.*

Choice rules (a special type of a logical formula) allows to make nondeterministic choices. For example the rule below chooses at least 1 and at most 5 intern among candidates, for each lab:

$$1 \leq \{\textit{intern(L,C)} : \textit{candid(C)}\} \leq 5 \leftarrow \textit{lab}(L). \tag{2}$$

(Hard) constraint rules eliminate those answer sets which do not qualify as solution. For example the constraint below prohibits the cases where a candidate is assigned to multiple labs:

$$\leftarrow\ L1 \neq L2,\ \textit{intern}(L1,C), \textit{intern}(L2,C). \tag{3}$$

Weak constraint rules express preferences and are used for optimization. The first rule below adds 2 unit of cost (penalty) for each intern and course, if an intern has not taken a preliminary course; and the priority of this rule is 1 (highest). The ASP solver tries to minimize the total cost in a lexicographic manner, starting from the highest priority (1).

$$\leftarrow\ \textit{intern}(L,I),\ \textit{not taken}(I,Z),\ \textit{prelim}(Z).[2@1,I,Z] \tag{4}$$

Further information about ASP can be found in [Baral2003, Gebser *et al.*2012, Gelfond and Kahl2014, Lifschitz2019].

3 Reasoning with $HOPRA$

Note that $HOPRA$ already includes *OPRA* relations, in addition it also includes qualitative and quantitative constraints about orientation, distance and location. *OPRA* relations are defined over the continuous space $\mathbb{R}^2 \times [0, 360^o)$. However instantiation and search for spatial variables over continuous domain is not feasible, besides the real data from measurement devices has finite precision and location of an extended object (like robot, ship) can be determined within some bounds. To examplify, a robot or a ship is an extended object with a long deck, its center point is always on the water and cannot reach the port. Hence on a continuous domain, we can never have "the ship is at the port". Thereby we define semantics of $HOPRA$ relations over the discrete space. Let $\Xi_{s,t} = [x_1, ..., x_s] \times [y_1, ..., y_t]$ be equally spaced x and y coordinate values and $\Phi_z = [f_1, ..., f_z]$ be the angular degree values. For example, if the range in both axis is 8 km, the precision is 0.1 km and angular resolution is 2^o, then $\Xi = [0, 0.1, 0.2, ..., 8.0]^2$ and $\Phi = [0, 2^o, 4^o, ..., 358^o]$. We can further represent the discrete domain by a grid such that $\Xi_{s,t} = [1, ..., s] \times [1, ..., t]$ and $\Phi_z = [1, ..., z]$. Then the domain of $HOPRA$ relations become $D_{s,t,z} = \Xi_{s,t} \times \Phi_z$.

We say that a given set C of $HOPRA$ constraints are consistent over domain $D_{s,t,z}$ if there is an instantiation of variables $E : V \mapsto D_{s,t,z}$ which satisfies constraints in C, and we call E a solution of C. We define the consistency checking problem H in $HOPRA$ as deciding whether the input network C is consistent or not.

Consistency problem of *OPRA* calculus itself is *NP*-hard and $\exists\mathbb{R}$-complete [Lee2014], and it is not in *NP* unless these complexity classes coincide. Theorem 1 states that complexity of consistency checking in $HOPRA$ is NP-complete. The reason for *OPRA* having higher complexity is that it is defined over continuous space: A nondeterministic Turing machine may not search over an infinite space.
Theorem 1. *The consistency checking problem $H = (C, V, D_{s,t,z})$ in HOPRA is NP-complete.*

4 Reasoning with $HOPRA$ using ASP

We first study consistency checking problem $H = (C, V, D_{s,t,z})$ in $HOPRA$ using Answer Set Programming. Due to limited space, we explain the core ASP rules, the full ASP code is available in the supplementary material.

Represent the input.

The input of the problem H is represented by a set of facts in ASP. An oriented spatial object $v \in V$ is represented by *object*(v) atom. In some domains (like surveying and our robotic scenario), it is known that two objects are located at the same point (with or without knowing their numerical coordinates). Hence we distinguish points and objects; and denote point identifier of an object with another atom *point*(v, p). v and p are just variables and we enumerate them with $1..|V|$ and $1..|P|$. In general, if point location of objects are unknown, then their point identifier can be chosen the same as the object identifier.

We encode *OPRA* constraints by *same_opra(N,K,L,I,G)*, *diff_opra(N,K,L,I,J,G)*, *one_sided_diff_opra(N,K,L,I,G)* atoms. Here N denotes the constraint number, K, L objects, I, J sectors, G granularity. Qualitative distance relation is encoded by *dist_rel(N,K,L,U)* where U is the relation between K, L. We represent the quantitative constraints by *direction(N,K,A)*, *direction_range(N,K,A1,A2)*, *direction(N,K,P,R)*, *psi_precise(N,K,L,A)*, *psi_range(N,K,L,A1,A2)*, *delta_precise(N,K,L,A)*, *delta_range(N,K,L,A1,A2)*, *location(N,K,X,Y* *distance(N,K,L,D)*, *distance_range(N,K,L,D1,D2)* atoms. Because a differential OPRA constraint involves two different relative position information, we transform it into two single-sided constraint.

$$\begin{array}{l} \textit{single\_diff\_opra(N,K,L,I,G)} \leftarrow \textit{diff\_opra(N,K,L,I,J,G)}. \\ \textit{single\_diff\_opra(N,L,K,J,G)} \leftarrow \textit{diff\_opra(N,K,L,I,J,G)}. \\ \textit{single\_diff\_opra(N,K,L,I,G)} \leftarrow \textit{one\_sided\_diff\_opra(N,K,L,I,G)}. \end{array} \quad (5)$$

single_diff_opra(N,K,L,I,G) reads as K is on sector I of L with granularity G. We process quantitative constraints to identify location and orientation (heading) of objects.

$$\begin{array}{l} \textit{xloc(P,X)} \leftarrow \textit{point(K,P)},\ \textit{location(N,K,X,Y)}. \\ \textit{yloc(P,Y)} \leftarrow \textit{point(K,P)},\ \textit{location(N,K,X,Y)}. \\ \textit{orient(K,A)} \leftarrow \textit{direction(N,K,A)}. \\ \textit{loc\_known(P)} \leftarrow \textit{point(K,P)},\ \textit{location(N,K,X,Y)}. \\ \textit{orient\_specified(K)} \leftarrow \textit{direction(N,K,A)}. \\ \textit{dist\_known(P,R)} \leftarrow \textit{point(K,P)},\ \textit{point(L,R)}, \\ \qquad \textit{distance(N,K,L,D)}. \end{array} \quad (6)$$

Disjunctive Constraints. We represent uncertainity in quantitative constraints by a range of values. Uncertainity in qualitative OPRA and distance constraints are represented by disjunctive constraints of the form *disjrelation*$(N, D, Y, ..)$ where N is the constraint ID, D is the disjunct ID, Y is the type of the disjunctive constraint and the rest are OPRA and distance constraint parameters. We enumerate the type of the disjunctive constraint as 0:only differential OPRA constraint, 1:only same OPRA constraint, 2:differential OPRA plus distance constraint, 3:only one-sided differential OPRA constraint, 4:one-sided differential OPRA constraint plus distance, 5:only distance constraint.

If the N^{th} constraint is disjunctive the rule below nondeterministically picks one disjunct and *chosen*(N, D) atom indicates its index.

$$\{\textit{chosen(N,D)} :\ \textit{disj\_index(N,D)}\} = 1 \ \leftarrow \ \textit{existdisj}(N). \quad (7)$$

Then we generate the OPRA and/or distance constraint corresponding to the selected disjunct. As an example, the rule for type 2 is given below.

$$\begin{array}{l} \textit{diff\_opra(N,K,L,I,J,G)} \leftarrow \textit{chosen(N,D)}, \\ \qquad \textit{disjrelation(N,D,2,K,L,I,J,G,U)}. \\ \textit{dist\_rel(N,K,L,U)} \leftarrow \textit{chosen(N,D)}, \\ \qquad \textit{disjrelation(N,D,2,K,L,I,J,G,U)}. \end{array} \quad (8)$$

Instantiation of spatial variables. A candidate solution is characterized by an instantiation of objects $u \in V$ by an oriented point in $D_{s,t,z}$. An oriented point is specified by its location and orientation. We nondeterministically generate a location $(x, y) \in \Xi_{r,s}$ on the grid for those points whose location is unknown. Forbidden locations and obstacles are designated by *invalid_loc*(X, Y) atoms in the input.

$$\begin{array}{l} \{\textit{xloc(P,X)} :\ 0 \le X < r\} = 1 \ \leftarrow \textit{not loc\_known(P)},\ \textit{point(P)}. \\ \{\textit{yloc(P,Y)} :\ 0 \le Y < s\} = 1 \ \leftarrow \textit{not loc\_known(P)},\ \textit{point(P)}. \\ \leftarrow \ \textit{invalid\_loc}(X,Y),\ \textit{xloc}(P,X),\ \textit{yloc}(P,Y). \end{array} \quad (9)$$

We nondeterministically assign an orientation (heading) $\phi \in \Phi_z$ to the objects whose orientation is unknown. To examplify, if angular resolution is 3^o, then possible values are $\{0, 3, 6, ..., 357\}$. For this, we generate a value in $\Omega = \{0, 1, 2, ..., 119\}$ and multiply by 3.

$$\begin{array}{l} \{\textit{degree}(K,E) :\ E \in \Omega\} = 1 \ \leftarrow \\ \qquad \textit{not orient\_specified}(K),\ \textit{point(P)}. \\ \textit{orient}(K, E.Z) \ \leftarrow \ \textit{degree}(K,E),\ \textit{angle\_res}(Z), \\ \qquad \textit{not orient\_specified}(K),\ \textit{point(P)}. \end{array} \quad (10)$$

Next we determine which vectors and angles to compute for OPRA and quantitative constraints. We compute the numerical distance between coordinates and find the angle φ_{PR} by arctangent. We represent *tan* function by internal ASP atoms for example $\mathit{tan\_range}(4, 0.062, 0.078)$ shows the range of 4^o. For differential OPRA constraints and quantitative ψ constraints, we need to compute the vectors $\overrightarrow{PR}$, $\overrightarrow{RP}$ and the angle of these vectors ϕ_{PR}, ϕ_{RP}.

$$\begin{array}{l}
\mathit{compute\_vector(P,R)} \leftarrow \mathit{point(K,P)},\ \mathit{point(L,R)},\\
\quad \mathit{diff\_opra(N,K,L,I,J,G)}.\\
\mathit{xdist(P,R,X1\text{-}X2)} \leftarrow \mathit{xloc(P,X1)},\ \mathit{xloc(R,X2)},\\
\quad \mathit{compute\_vector(P,R)}.\\
\mathit{ydist(P,R,Y1\text{-}Y2)} \leftarrow \mathit{yloc(P,Y1)},\ \mathit{yloc(R,Y2)},\\
\quad \mathit{compute\_vector(P,R)}.\\
\mathit{tan}(P,R,DY/DX) \leftarrow DX \neq 0,\\
\quad \mathit{xdist(P,R,DX)},\ \mathit{ydist(P,R,DY)}.\\
\mathit{varphi}(P,R,A) \leftarrow B \geq T1,\ B \leq T2,\\
\quad \mathit{tan(P,R,B)},\ \mathit{tan\_range(A,T1,T2)}.
\end{array} \tag{11}$$

OPRA Constraints. We add ASP rules to impose and process differential and same OPRA constraints.

Firstly, if two objects are related a same OPRA constraint then their locations must be identical. Thus location of one of them can be deduced from the other, without loss of generality we calculate the lower-indexed one.

$$\begin{array}{l}
\mathit{xloc}(P,X) \leftarrow P < R,\ \mathit{xloc}(R,X),\\
\quad \mathit{point}(K,P),\ \mathit{point}(L,R),\ \mathit{same\_opra(N,K,L,I,G)}.\\
\mathit{xloc}(R,X) \leftarrow P > R,\ \mathit{xloc}(P,X),\\
\quad \mathit{point}(K,P),\ \mathit{point}(L,R),\ \mathit{same\_opra(N,K,L,I,G)}.\\
\mathit{yloc}(P,Y) \leftarrow P < R,\ \mathit{yloc}(R,Y),\\
\quad \mathit{point}(K,P),\ \mathit{point}(L,R),\ \mathit{same\_opra(N,K,L,I,G)}.\\
\mathit{yloc}(R,Y) \leftarrow P > R,\ \mathit{yloc}(P,Y),\\
\quad \mathit{point}(K,P),\ \mathit{point}(L,R),\ \mathit{same\_opra(N,K,L,I,G)}.\\
\mathit{loc\_known}(P) \leftarrow P < R,\ \mathit{point}(K,P),\ \mathit{point}(L,R),\\
\quad \mathit{same\_opra(N,K,L,I,G)}.\\
\mathit{loc\_known}(R) \leftarrow P > R,\ \mathit{point}(K,P),\ \mathit{point}(L,R),\\
\quad \mathit{same\_opra(N,K,L,I,G)}.
\end{array} \tag{12}$$

For a differential OPRA constraint, if the sector is linear (even indexed), the orientation of object L can be calculated from *varphi(P,R,A)*. Likewise, for a linear same OPRA constraint, orientation of one object (the lower indexed one) can be calculated from the other.

$$\begin{array}{l}
\mathit{orient}(L,(A - 90.I/G)\%360) \leftarrow I \,\%\, 2 = 0,\ \mathit{varphi(P,R,A)},\\
\quad \mathit{point(K,P)},\ \mathit{point(L,R)},\ \mathit{single\_diff\_opra(N,K,L,I,J,G)}.\\
\mathit{orient}(K,(T - I.90/G)\%360) \leftarrow I \,\%\, 2 = 0,\ L > K,\\
\quad \mathit{orient}(L,T),\ \mathit{same\_opra}(N,K,L,I,G).\\
\mathit{orient}(L,(T + I.90/G)\%360) \leftarrow I \,\%\, 2 = 0,\ K > L,\\
\quad \mathit{orient}(K,T),\ \mathit{same\_opra}(N,K,L,I,G).\\
\mathit{orient\_specified}(L) \leftarrow I \,\%\, 2 = 0,\ \mathit{single\_diff\_opra(N,K,L,I,J,G)}.\\
\mathit{orient\_specified}(K) \leftarrow I \,\%\, 2 = 0,\ L > K,\ \mathit{same\_opra(N,K,L,I,G)}.\\
\mathit{orient\_specified}(L) \leftarrow I \,\%\, 2 = 0,\ K > L,\ \mathit{same\_opra(N,K,L,I,G)}.
\end{array} \tag{13}$$

For same OPRA constraints with angular sectors, we use the rules below to check them.

$$\begin{array}{l}\leftarrow\ I\,\%\,2=1,\ (T-B)\,\%\,360\leq(I-1).90/G,\ \textit{orient(K,B)},\\ \quad \textit{orient(L,T)},\ \textit{same\_opra(N,K,L,I,G)}.\\ \leftarrow\ I\,\%\,2=1,\ (T-B)\,\%\,360\geq(I+1).90/G,\ \textit{orient(K,B)},\\ \quad \textit{orient(L,T)},\ \textit{same\_opra(N,K,L,I,G)}.\end{array}\tag{14}$$

For an angular differential OPRA constraint, the guessed value of ϕ_L must be compatible with φ_{LK}. For example, if $\varphi_{LK}=250^o$ and $I=3$, $G=4$ then each angular segment is 45^o and ϕ_L must be in the range $(250-2\times45^o, 250-45^o)=(160^o,205^o)$. We must also handle the case when bounds are reverse (i.e. change due to modulo 360). For example, if $\varphi_{LK}=280^o$ and $I=13$, $G=4$ then ϕ_L must be in the range $(280-7\times45^o,280-6\times45^o)=(325^o,10^o)$. In this case, ϕ_L must not be less than 325 and greater than 10.

$$\begin{array}{l}\leftarrow\ I\,\%\,2=1,\ T\leq(A-(I+1).90/G)\,\%\,360,\\ \quad (A-(I-1).90/G)\,\%\,360>(A-(I+1).90/G)\,\%\,360,\\ \quad \textit{orient(L,T)},\ \textit{varphi(P,R,A)},\ \textit{point(K,P)},\\ \quad \textit{point(L,R)},\ \textit{single\_diff\_opra(N,K,L,I,G)}.\\ \leftarrow\ I\,\%\,2=1,\ T\geq(A-(I-1).90/G)\,\%\,360,\\ \quad (A-(I-1).90/G)\,\%\,360>(A-(I+1).90/G)\,\%\,360,\\ \quad \textit{orient(L,T)},\ \textit{varphi(P,R,A)},\ \textit{point(K,P)},\\ \quad \textit{point(L,R)},\ \textit{single\_diff\_opra(N,K,L,I,G)}.\\ \leftarrow\ I\,\%\,2=1,\ T\leq(A-(I+1).90/G)\,\%\,360,\\ \quad T\geq(A-(I-1).90/G)\,\%\,360,\\ \quad (A-(I-1).90/G)\,\%\,360<(A-(I+1).90/G)\,\%\,360,\\ \quad \textit{orient(L,T)},\ \textit{varphi(P,R,A)},\ \textit{point(K,P)},\\ \quad \textit{point(L,R)},\ \textit{single\_diff\_opra(N,K,L,I,G)}.\end{array}\tag{15}$$

Quantitative Constraints. We process quantitative $HOPRA$ constraints to restrict intrinsic orientation and location of objects, using a similar method in OPRA constraints. For example, the ASP rules below handle the $\phi(K)$, $\psi(K,L)$, $\delta(K,L)$ constraints. In particular, in the case of a precise $\psi(K,L)$, $\delta(K,L)$ constraint, orientation of one object can be deduced from the other. $\psi(K,L)$, $\delta(K,L)$ range constraints can be imposed by ASP constraints (e.g. the last two rules).

$$\begin{array}{l}\leftarrow T<A1,\ \textit{orient(K,T)},\ \textit{direction\_range(N,K,A1,A2)}.\\ \leftarrow T>A2,\ \textit{orient(K,T)},\ \textit{direction\_range(N,K,A1,A2)}.\\ \textit{orient\_specified(L)}\leftarrow \textit{psi\_precise(N,K,L,T)}.\\ \textit{orient\_specified}(K)\leftarrow L>K,\ \textit{delta\_precise(N,K,L,A)}.\\ \textit{orient\_specified}(L)\leftarrow K>L,\ \textit{delta\_precise(N,K,L,A)}.\\ \textit{orient}(L,(A-T)\%360)\leftarrow \textit{varphi(P,R,A)},\\ \quad \textit{point(K,P)},\ \textit{point(L,R)},\ \textit{psi\_precise(N,K,L,T)}.\\ \textit{orient}(K,(T-A)\%360)\leftarrow\ L>K,\ \textit{orient(L,T)},\ \textit{delta\_precise(N,K,L,A)}.\\ \textit{orient}(L,(T+A)\%360)\leftarrow\ K>L,\ \textit{orient(K,T)},\ \textit{delta\_precise(N,K,L,A)}.\\ \leftarrow\ (T-B)\%360<A1,\ \textit{orient(K,B)},\ \textit{orient(L,T)},\ \textit{delta\_range(N,K,L,A1,A2)}.\\ \leftarrow\ (T-B)\%360>A2,\ \textit{orient(K,B)},\ \textit{orient(L,T)},\ \textit{delta\_range(N,K,L,A1,A2)}.\end{array}\tag{16}$$

Distance Constraints. For a qualitative distance constraint, we ensure that the numerical distance (square) between objects is inside its interval. For a quantitative distance constraint, we check whether the generated coordinates satisfy the numerical distance.

$$\begin{array}{l}\textit{ndist}(P,R,X^2+Y^2)\leftarrow \textit{xdist}(P,R,X),\ \textit{ydist}(P,R,Y).\\ \leftarrow D<D1,\ \textit{lower\_bound(U,D1)},\ \textit{ndist(P,R,D)},\\ \quad \textit{point(K,P)},\ \textit{point(L,R)},\ \textit{dist\_rel(N,K,L,U)}.\\ \leftarrow D>D2,\ \textit{upper\_bound(U,D2)},\ \textit{ndist(P,R,D)},\\ \quad \textit{point(K,P)},\ \textit{point(L,R)},\ \textit{dist\_rel(N,K,L,U)}.\\ \leftarrow\ Y^2\neq D^2-X^2,\ \textit{xdist(P,R,X)},\ \textit{ydist(P,R,Y)},\\ \quad \textit{point(K,P)},\ \textit{point(L,R)},\ \textit{distance(N,K,L,D)}.\end{array}\tag{17}$$

Henceforth the ASP program Π for checking consistency of $HOPRA$ constraints is composed of the rules (5)-(17) and the facts that represent the input. An answer set of the program Π specifies a configuration of oriented objects in the grid, identified by $\mathit{xloc}(P, X)$, $\mathit{yloc}(P, Y)$, $\mathit{orient}(K, A)$ atoms. Theorem 2 states that Π is a sound and complete solution for H over domain $D_{s,t,z}$. However, if the resolution parameters s, t, z are less than the sensor precision (that constraints are measured) i.e. the grid size is smaller than actual, then theoretically Π may not yield an answer set though constraints are satisfiable (this never happened in our experiments).

Theorem 2. *Let $H = (C, V, D_{s,t,z})$ be an $HOPRA$ consistency problem and $\mathcal{O}_{s,t,z}$ denote the set of all $\mathit{xloc}(P, X)$, $\mathit{yloc}(P, Y)$, $\mathit{orient}(K, A)$, $\mathit{point}(K, P)$ atoms. An assignment X from $D_{s,t,z}$ to objects in V is a solution of H if and only if X can be represented as $X = Z \cap \mathcal{O}_{s,t,z}$ for some answer set Z of Π. Moreover, every solution of H can be represented in this form in only one way.*

5 Guidance Mechanism for Solution Search

Previous on constraint-based reasoning suggests that adding some redundant constraints may be beneficial for speeding up the search process [Barták2001, Cheng *et al.*1999, Karwan *et al.*2012, van Emden1999]. The additional constraints guide the search engine to eliminate the invalid candidates/choices for variables and reach a solution on the search tree faster. In order to improve the computation time for $HOPRA$ (especially for inconsistent problem instances), we have developed an ASP subprogram with redundant constraints to guide the search. This subprogram is oriented towards/designed for pruning candidate solutions for angular differential OPRA constraints and quantitative ψ and distance constraints, since these constraints are harder to check and satisfy. To our knowledge, this type of guidance mechanism is also new in the spatial reasoning literature.

We first guide nondeterministic choice of objects' orientation using differential OPRA constraints. If there are two single-sided differential OPRA constraints for a symmetric pair i.e. $K\ {}_{g_1}\measuredangle^{i}\ L$ and $L\ {}_{g_2}\measuredangle^{j}\ K$, then the difference between orientation of objects $\phi_L - \phi_K + 180$ must lie between $(\frac{90.(j-1)}{g_2} - \frac{90.(i+1)}{g_1})$ and $(\frac{90.(j+1)}{g_2} - \frac{90.(i-1)}{g_1})$. Note that we must handle the case of reverse bounds separately (i.e. change due to modulo 360).

$$
\begin{aligned}
\leftarrow\ & I\,\%\,2\!=\!1,\ J\,\%\,2\!=\!1,\ K\!>\!L,\\
& (B-T+180)\,\%\,360 \le ((J-1).90/G_2) - ((I+1).90/G_1)\,\%\,360,\\
& ((J-1).90/G_2) - ((I+1).90/G_1)\,\%\,360 \le\\
& ((J+1).90/G_2) - ((I-1).90/G_1)\,\%\,360,\\
& \mathit{orient(K,T)},\ \mathit{orient(L,B)},\\
& \mathit{single\_diff\_opra}(M,K,L,I,G_1),\ \mathit{single\_diff\_opra}(N,L,K,J,G_2).\\
\leftarrow\ & I\,\%\,2\!=\!1,\ J\,\%\,2\!=\!1,\ K\!>\!L,\\
& (B-T+180)\,\%\,360 \ge ((J+1).90/G_2) - ((I-1).90/G_1)\,\%\,360,\\
& ((J-1).90/G_2) - ((I+1).90/G_1)\,\%\,360 \le\\
& ((J+1).90/G_2) - ((I-1).90/G_1)\,\%\,360,\\
& \mathit{orient(K,T)},\ \mathit{orient(L,B)},\\
& \mathit{single\_diff\_opra}(M,K,L,I,G_1),\ \mathit{single\_diff\_opra}(N,L,K,J,G_2).\\
\leftarrow\ & I\,\%\,2\!=\!1,\ J\,\%\,2\!=\!1,\ K\!>\!L,\\
& (B-T+180)\,\%\,360 \le ((J-1).90/G_2) - ((I+1).90/G_1)\,\%\,360,\\
& (B-T+180)\,\%\,360 \ge ((J+1).90/G_2) - ((I-1).90/G_1)\,\%\,360,\\
& ((J-1).90/G_2) - ((I+1).90/G_1)\,\%\,360 >\\
& ((J+1).90/G_2) - ((I-1).90/G_1)\,\%\,360,\\
& \mathit{orient(K,T)},\ \mathit{orient(L,B)},\\
& \mathit{single\_diff\_opra}(M,K,L,I,G_1),\ \mathit{single\_diff\_opra}(N,L,K,J,G_2).
\end{aligned}
\tag{18}
$$

Next, we guide the generation of object location using differential OPRA and quantitative $HOPRA$ constraints. Consider a single-sided OPRA constraint $K\ {}_{g}\measuredangle^{i}\ L$ in the input. Based on the chosen value of ϕ_L, if the allowed x (or y) coordinate of object K, required by this OPRA constraint is to the right/left (or up/below) of object L, then we impose the difference between x (or y) coordinates to be positive/negative accordingly.

$$\begin{aligned}
&\leftarrow\ I \,\%\, 2 = 1,\ Y \leq 0,\ \mathit{ydist(K,L,Y)},\ (T + (I-1).90/G) \,\%\, 360 \leq 180,\\
&\qquad (T + (I+1).90/G) \,\%\, 360 \leq 180,\\
&\qquad \mathit{orient(L,T)},\ \mathit{single\_diff\_opra(N,K,L,I,G)}.\\
&\leftarrow\ I \,\%\, 2 = 1,\ Y \geq 0,\ \mathit{ydist(K,L,Y)},\ (T + (I-1).90/G) \,\%\, 360 \geq 180,\\
&\qquad (T + (I+1).90/G) \,\%\, 360 \geq 180,\\
&\qquad \mathit{orient(L,T)},\ \mathit{single\_diff\_opra(N,K,L,I,G)}.
\end{aligned} \tag{19}$$

and analogous rules for x axis. Using a similar idea, we guide generation of object location using quantitative $\psi(K, L)$ and distance constraints. Namely, if a precise ψ or range ψ constraint requires object K to be located right/left (or up/below) of object L, then the set of rules below ensures that the x (or y) coordinate of K is greater/less than L. In case of a distance constraint $\mathit{dist(K,L)} = c$, we ensure that the distance over x or y axis cannot be greater than c.

$$\begin{aligned}
&\leftarrow\ Y < 0,\ \mathit{ydist(K,L,Y)},\ (T + A) \,\%\, 360 \leq 180,\\
&\qquad \mathit{orient(L,T)},\ \mathit{psi\_precise(N,K,L,A)}.\\
&\leftarrow\ Y > 0,\ \mathit{ydist(K,L,Y)},\ (T + A) \,\%\, 360 \geq 180,\\
&\qquad \mathit{orient(L,T)},\ \mathit{psi\_precise(N,K,L,A)}.\\
&\leftarrow\ Y < 0,\ \mathit{ydist(K,L,Y)},\\
&\qquad (T + A1) \,\%\, 360 \leq 180,\ (T + A2) \,\%\, 360 \leq 180,\\
&\qquad \mathit{orient(L,T)},\ \mathit{psi\_range(N,K,L,A1,A2)}.\\
&\leftarrow\ Y > 0,\ \mathit{ydist(K,L,Y)},\\
&\qquad (T + A1) \,\%\, 360 \geq 180,\ (T + A2) \,\%\, 360 \geq 180,\\
&\qquad \mathit{orient(L,T)},\ \mathit{psi\_range(N,K,L,A1,A2)}.\\
&\leftarrow\ |Y| > 0,\ \mathit{ydist(K,L,Y)},\\
&\qquad \mathit{point(K,P)},\ \mathit{point(L,R)},\ \mathit{distance(N,K,L,D)}.
\end{aligned} \tag{20}$$

and analogous rules for x axis.

Then the improved ASP program Π^i for $HOPRA$ with guidance mechanism is formed by the union of the original ASP program Π with the rules (18)-(20).

5.1 Inferring New Knowledge

If the given information is incomplete, agents may need to deduce new knowledge. The desired relations to infer can be specified at the input by $\mathit{toinfer\_opra}(K, L, G)$, $\mathit{toinfer\_distrel}(K, L)$, ... type atoms. The unknown relations can be inferred from the generated location and orientation of objects in the answer set. For example,

$$\begin{aligned}
&\mathit{inferred\_orient(K,A)} \leftarrow \mathit{orient(K,A)},\ \mathit{toinfer\_orient(K)}\\
&\mathit{inferred\_same\_opra}(K, L, Z/(90/G), G) \leftarrow\\
&\qquad Z\%(180/G) = 0,\ \mathit{orient\_diff}(K, L, Z),\\
&\qquad \mathit{same\_loc(K,L)}, \mathit{toinfer\_opra(K,L,G)}.\\
&\mathit{inferred\_distrel(K,L,U)} \leftarrow D \geq D1,\ D \leq D2,\\
&\qquad \mathit{upper\_bound}(U, D1),\ \mathit{upper\_bound}(U, D2),\\
&\qquad \mathit{ndist}(P, R, D),\ \mathit{point}(K, P),\ \mathit{point}(L, R),\ \mathit{toinfer\_distrel}(K, L).
\end{aligned} \tag{21}$$

Note that there may be multiple configuration of objects that satisfy the given constraints and the inferred relations may not be unique. All possible inferred relations can be obtained by computing all answer sets by the ASP solver.

5.2 Explaining the Source of Inconsistency

It might be the case that the $HOPRA$ constraints in C are inconsistent with each other, i.e. C is unsatisfiable. However, when we exclude some constraints from C, it may become satisfiable. In that sense, the set of excluded constraints explain a source of inconsistency in the original set C. To identify the source of inconsistency, we nondeterministically select which constraints to drop. $drop(N)$ atom indicates that the constraint with ID number N is dropped. The mandatory (non-defeasible) constraints are specified by $mandatory(N)$ atom in the input. Each dropped constraint adds 1 unit cost and the ASP solver tries to minimize the total cost. We also revise the ASP rules so that the dropped constraints do not apply.

$$\begin{array}{l} \{drop(N)\} \leq 1 \leftarrow \text{not } mandatory(N),\ constraint\_id(N). \\ \overset{\sim}{\leftarrow}\ drop(N) \quad [1@1, N]. \end{array} \tag{22}$$

5.3 Presumed Information

In some situations, agents might have prior or presumed information, for example the port station knows that ship 4 planned to sail from Izmir port to Gocek Marina, so its heading is presumably towards Gocek. We call these presumed $HOPRA$ constraints. The reasoning agent should consider these (qualitative or quantitative) presumed constraints unless they conflict with the existing $HOPRA$ constraints. In this sense the presumed constraints are defeasible. We represent presumed contraints by $presumed\_diff\_opra(N,K,L,I,J,G)$, $presumed\_same\_opra(N,K,L,I,G)$, $presumed\_orient(N,K,A)$ and similar atoms. We minimize the number of violated presumed constraints by

$$\begin{array}{l} \{violated(N)\} \leq 1 \ \leftarrow\ presumed\_constraint\_id(N). \\ \overset{\sim}{\leftarrow}\ violated(N) \quad [1@2, N]. \end{array} \tag{23}$$

The priority of the weak constraint in rule (22) is higher than that of rule (23) because satisfiability of original $HOPRA$ constraints is more important. If the presumed constraint applies, the corresponding atoms are generated. For example,

$$\begin{array}{l} diff\_opra(N,K,L,I,J,G) \ \leftarrow\ \text{not } violated(N), \\ \qquad presumed\_diff\_opra(N,K,L,I,J,G). \end{array} \tag{24}$$

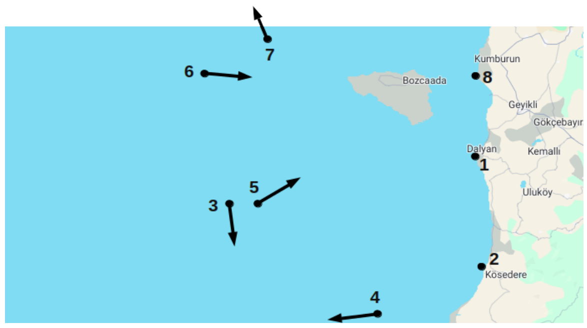
6 Applications of $HOPRA$

6.1 Marine Navigation:

We first consider fleet management of a group of six vessels sailing in the Agean Sea and the port station at Dalyan. The ships report the following data to the station Vessel 4 observes vessel 2 near and on its starboard 147^o and 2 is staying motionless at Kosedere marina. 5 reports its location is $(18, 11)$ and he observes 4 on its right-front. Vessel 6 reports that it is distant from the station by 20-25 km. Vessel 3 states that 5 is on its port direction between $70^o - 100^o$, and 6 is on his behind and not near. 7 observes 6 on his clock angle 9 and very near to it.

The agent at port station also knows that vessel 6 had a plan to go to Kumburun port. Based on the collected information, the port station wants to (1) verify that all measured data are correct and (2) find out the orientation of vessel 3 and its position wrt 4. For this, the station agent uses the main ASP program augmented with subprograms for inference and presumed constraints. The grid is 40x30 with angular resolution 3^o and each slot 1km. The combined program yields an answer set (Fig. 3), and from the *inferred_orient*$(3, 330)$, *inferred_diff_opra*$(3, 4, 1, 4)$ atoms, the agent infers that 3 is moving 330^o and left-front of 4.

Figure 3: ASP output of marine navigation scenario



6.2 Cognitive Robotics:

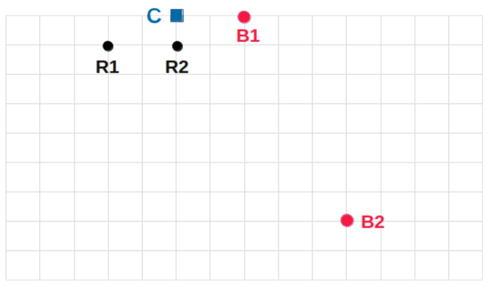
Next we study the robotic perception problem [Moratz *et al.*2005, Dylla and Moratz2004, Moratz and Ragni2008], mentioned in the introduction. The task of the robot is to navigate to the red cube behind the blue cube. To achieve this, the robot first needs to identify which red cube is behind the blue cube (C). In this mission, the robot makes two preceptions at point R1 and R2 with its sensor. The sensor data at these points are stated as OPRA relations in [Moratz *et al.*2005] as below. Here P_R denotes the object located at point P, oriented toward R and $\,_R\, P$ denotes the object located at point P, whose orientation is vector $\overrightarrow{RP}$.
$(R1)\ R1_{R2}\ {}_4\angle 1\ R1_C\ ,\ \ R1_{R2}\ {}_4\angle 1\ R1_{B1}\ ,\ \ R1_{R2}\ {}_4\angle 15\ R1_{B2}$
$(R2)\ R1_{R2}\ {}_4\angle_0^8\ {}_{R1}\ R2\ ,\ \ {}_{R1}\ R2\ {}_4\angle 4\ R2_C\ ,$
${}_{R1}\ R2\ {}_4\angle 1\ R2_{B1}\ ,\qquad {}_{R1}\ R2\ {}_4\angle 13\ R2_{B2}$

Based on preceived information, the robot must determine which object(s) are behind the cube i.e. check whether the following (disjunctive) constraints hold: $B1\ {}_4\measuredangle^{\{7,8,9\}}\ C_{R1}\ ,\qquad B2\ {}_4\measuredangle^{\{7,8,9\}}\ C_{R1}$
One method to solve this problem is inference as in the first scenario, an alternative method we present here is checking (in)consistency. We impose the physical sensor data as mandatory OPRA constraints; we add the candidate relations above as non-mandatory constraints. Then we utilize the main ASP program together with the subprogram for inconsistency which tries to satisfy the maximum number of constraints. At an optimal answer set, the first disjunctive constraint is satisfied but the second is not. Thus object B1 is behind C but B2 cannot be. The layout of objects in this solution is shown below. With composition and inversion based reasoning, [Moratz *et al.*2005] could not reject that B2 is behind C. This constitutes another example that path-consistency type approaches are insufficient for *OPRA*. Thus our model is better suited for checking consistency and reasoning with *OPRA* and $HOPRA$. The setup of this scenario is also relevant in other applications such as surveying, mapping, landmark localization. An example of surveying street networks using *OPRA* is [Lücke *et al.*2011].

7 Experimental Evaluations

We perform experiments to assess computational efficiency of our ASP formulation and observe the impact of input size, granularity, grid and type of constraints. All tests performed on a Linux workstation with Intel i9-9900K 3.6GHz CPU, 64GB memory using the ASP solver Clingo 5.4.0 on a single core. We

Figure 4: ASP output of robotic perception and motion scenario



first experimented with basic *OPRA* constraints and original ASP program Π to evaluate performance of consistency checking in *OPRA*. For a number of object (N) and constraint (C), we have created 100 consistent and 100 inconsistent random benchmark instances (samples). We set granularity (G) uniform in an instance. Consistent problem instances are created by randomly picking location and heading of objects on a 1000x1000 grid. The pair of objects in the constraint set are randomly chosen and *OPRA* constraints are extracted from the layout. Inconsistent instances are created by assigning random *OPRA* relations to those object pairs. We fixed the timeout value to 100 seconds and recorded time statistics (in seconds) over 100 samples: Grnd shows average grounding time while Mean, Max, Std shows average, maximum and std. deviation of solving time (grounding+search). [1] Tout denotes number of timeout instances. S, aR denotes the grid size is SxS and angular resolution is R^o.

Table 1: Results for basic *OPRA* constraints with original program Π

Instance				Consistent					Inconsistent				
N	C	G	Grid	Grnd	Mean	Max	Std	Tout	Grnd	Mean	Max	Std	Tout
10	10	4	50,a3	1.31	1.66	3.63	0.41	0	0.69	8.35	94.2	21.4	11
10	10	4	100,a2	2.39	5.59	33.2	4.83	0	1.25	2.32	32.5	4.34	27
10	10	6	50,a3	1.37	2.04	5.45	0.84	0	0.69	8.78	95.9	21.8	16
10	10	6	100,a2	2.47	6.97	37.9	5.10	0	1.23	3.59	81.8	9.66	32
10	20	4	50,a3	2.81	7.16	41.2	5.29	0	1.18	1.19	3.04	0.41	0
10	20	4	100,a2	5.09	35.7	92.6	23.6	21	2.13	2.16	6.44	0.79	0
20	20	2	50,a3	2.51	3.26	8.23	0.88	0	1.40	5.88	90.3	13.5	6
20	20	4	50,a3	2.88	5.21	23.9	3.36	0	1.52	6.71	95.7	16.0	12
20	40	2	50,a3	5.09	24.64	97.5	19.6	0	2.14	2.15	4.66	0.64	1
40	40	2	50,a3	5.31	9.90	33.9	4.88	0	2.80	7.63	87.8	16.2	13
60	60	2	25,a3	7.21	8.99	15.4	1.54	0	4.01	6.81	86.9	11.2	6

Note that these are (double-sided) differential *OPRA* constraints, each of them actually includes two constraints. We observe that grounding time and total computation time increase as the number of objects, constraints, granularity and grid size increase. In general, the computation time and timeout values for inconsistent instances are greater compared to consistent instances. This is mainly because it is sufficient to find one combination to reach consistent outcome whereas the solver needs to search many combinations on the search tree to decide inconsistency. Notice that the computation time for inconsistent instances are lower than consistent instances when the number of constraints or granularity is large. Our interpretation is that since constraints for inconsistent instances are randomly generated, it is relatively easier to detect conflicts between them, when the constraints are dense (number) or they are fine (granularity).

To assess the benefit of subprogram for guidance (in Section 5), we tested the same instances with ASP program Π^i. Comparing the results in Table 1 and 2 reveal that addition of subprogram for guidance increases grounding time a little (due to additional constraints), but the search is much faster, thus the net effect is lower computation time on average. Now inconsistency is determined faster than consistency since guidance rules makes conflict detection easier. Note that with guidance, the maximum computation

[1] Timed-out instances were not counted in calculting statistics in the tables

time and timeout instances are also lower, in general.

Table 2: Effect of Guidance: Basic *OPRA* constraints with Π^i

Instance				Consistent					Inconsistent				
N	C	G	Grid	Grnd	Mean	Max	Std	Tout	Grnd	Mean	Max	Std	Tout
10	10	4	50,a3	1.93	2.00	2.63	0.14	0	0.86	1.62	29.8	3.03	11
10	10	4	100,a2	4.09	4.50	5.78	0.43	0	1.75	4.10	86.4	9.39	14
10	10	6	50,a3	2.01	2.13	3.11	0.19	0	0.98	5.80	37.3	13.4	1
10	10	6	100,a2	4.25	4.80	9.46	0.79	0	1.83	3.73	35.5	6.65	15
10	20	4	50,a3	3.83	4.35	8.87	0.73	0	1.38	1.39	4.15	0.58	0
10	20	4	100,a2	7.82	10.7	35.2	3.73	0	2.69	2.84	10.5	1.60	0
20	20	2	50,a3	3.55	3.75	4.52	0.18	0	1.62	1.85	7.77	1.08	11
20	20	4	50,a3	4.09	4.39	7.04	0.35	0	1.94	2.49	12.0	1.94	5
20	40	2	50,a3	6.93	8.87	24.8	2.43	0	2.44	2.46	5.87	0.85	1
40	40	2	50,a3	7.41	8.19	10.6	0.41	0	3.34	4.92	37.9	5.04	10
60	60	2	25,a3	9.24	9.68	10.8	0.23	0	4.61	6.44	91.4	8.33	7
60	80	2	25,a3	12.4	14.8	31.9	2.87	0	5.32	5.80	37.7	3.90	1

Next we make experiments to observe the impact of qualitative distance constraints, we use the improved ASP program Π^i. For each pair, we add the distance constraint to the constraint set. For a consistent instance, we extract the distance relation from the layout and add to the constraint set. For inconsistent instance, we randomly pick the distance relation. In general, computation timings are higher compared to Table 2, due to additional overhead of checking distance constraints.

Table 3: Results for basic *OPRA* and qualitative distance constraints

Instance				Consistent					Inconsistent				
N	C	G	Grid	Grnd	Mean	Max	Std	Tout	Grnd	Mean	Max	Std	Tout
10	10	6	50,a3	2.40	2.66	4.12	0.30	0	1.06	5.85	67.8	13.6	6
10	20	4	50,a3	4.54	6.34	32.3	2.96	0	1.51	1.53	4.86	0.69	0
20	20	2	50,a3	4.13	4.91	13.0	1.12	0	1.87	10.1	97.0	21.5	12
20	20	4	50,a3	4.84	5.96	29.1	2.54	0	2.13	6.69	98.2	14.2	8
20	40	2	50,a3	8.27	22.0	76.8	14.6	6	2.70	2.73	8.08	1.10	1
40	40	2	50,a3	8.71	13.3	33.0	4.47	0	3.53	7.20	66.1	13.3	21

We also experiment with disjunctive *OPRA* constraints. We have used program Π^i and created problem instances as follows: We take a basic (in)consistent instance for a parameter combination N, C, G and convert $\%20$ or $\%40$ of the constraints into disjunctive constraints. $4x8$ denotes that there are 4 disjunctive constraints, each having 8 disjuncts. For the disjuncts, we keep the original basic relation and add other randomly generated basic *OPRA* relations as disjuncts. This manner we have created 100 consistent and 100 inconsistent instances with disjunctive constraints. In experiments, grid size is 50x50 and angular resolution is 3^o. With disjunctive constraints, computation time increases but not very large.

Table 4: Results for disjunctive *OPRA* constraints

Instance				Consistent					Inconsistent				
N	C	G	Disj	Grnd	Mean	Max	Std	Tout	Grnd	Mean	Max	Std	Tout
20	20	4	4x2	4.76	5.46	20.2	1.35	0	2.64	6.56	96.6	14.3	8
20	20	4	4x4	5.93	6.35	10.1	1.01	0	3.56	6.45	66.4	9.06	11
20	20	4	4x8	8.78	9.34	18.9	2.46	0	5.48	11.2	85.7	17.3	11
20	20	4	8x2	5.34	5.95	10.7	0.88	0	3.20	6.56	48.5	8.30	11
20	20	4	8x4	7.85	8.87	33.8	2.95	0	5.44	12.4	93.4	18.2	17
20	20	4	8x8	13.4	14.3	30.8	4.25	0	10.5	14.6	54.6	10.4	11
40	40	2	8x2	8.31	9.38	13.8	0.74	0	4.47	6.15	50.2	6.53	16
40	40	2	8x4	9.53	10.5	14.4	0.86	0	5.68	9.05	86.3	12.4	12
40	40	2	8x8	13.5	15.1	82.4	8.57	0	7.49	10.7	96.9	12.3	12
40	40	2	16x2	8.88	10.2	14.9	1.10	0	5.57	7.26	30.9	5.13	19
40	40	2	16x4	11.7	13.0	22.8	1.61	0	8.18	10.9	66.1	8.62	9
40	40	2	16x8	16.0	17.3	39.1	3.01	0	12.8	14.2	34.1	5.21	8

We have performed experiments for quantitative $HOPRA$ constraints, in particular orientation constraints $\phi(K) = a^o$. We have created problem instances as follows: We take a basic consistent or inconsistent instance with only OPRA constraints for a parameter combination N, C, G. Then for every object, we add its precise numerical orientation to the constraint set. For a consistent instance, we extract the numerical orientation from the layout and add to the constraint set. For inconsistent instance, we randomly assign an orientation in range [0,359]. The results in Table 5 are comparable and similar to Table 2. The reason is that, with quantitative information, the solver does not need to guess the orientation of objects; but now it has to find object locations in order to match to the exogenously given orientations, wrt OPRA relation sectors. These two effects seem to neutralize each other.

Table 5: Results for basic *OPRA* constraints with quantitative orientation constraints

Instance				Consistent					Inconsistent				
N	C	G	Grid	Grnd	Mean	Max	Std	Tout	Grnd	Mean	Max	Std	Tout
20	20	2	50,a3	3.67	3.87	4.57	0.22	0	1.77	5.82	82.8	17.3	11
20	20	4	50,a3	4.33	4.70	5.75	0.36	0	2.10	3.08	22.8	4.48	12
20	40	2	50,a3	7.47	9.86	24.1	3.34	0	2.63	2.69	6.44	1.13	0
20	40	4	50,a3	8.77	13.6	28.4	4.93	0	3.08	3.11	6.80	1.28	0
40	40	2	50,a3	7.72	8.58	9.79	0.46	0	3.23	3.88	9.54	2.16	3
40	40	4	50,a3	8.90	10.4	14.9	1.22	0	3.88	5.25	28.5	5.43	4

8 Related Literature

OPRA has been applied to robotic motion [Glez-Cabrera *et al.*2013], marine navigation [Dylla *et al.*2007], topological map learning [Wallgrün2010], surveying [Lücke *et al.*2011]. However reasoning with *OPRA* is NP-hard [Wolter and Lee2010] and currently there is no mechanism that can decide consistency of even basic *OPRA* constraints. Existing tools use composition and path-consistency for reasoning which is incomplete [Mossakowski and Moratz2010] and cannot generate configuration of objects.

Answer Set Programming has been applied to spatial reasoning [Baryannis *et al.*2018, Baryannis *et al.*2020, Brenton *et al.*2016, Li2012]. However these approaches are based on path-consistency and do not involve commonsense knowledge or assumptions. [Izmirlioglu and Erdem2023] has proposed a framework for reasoning with direction of extended objects using cardinal directions (north, east, southwest), but the objects are stationary and do not have an intrinsic orientation. Furthermore their setup is purely qualitative and does not include distance or location information. ASP has not yet been applied to *OPRA* or distance calculus.

9 Conclusion

Spatial reasoning about orientation and position is beneficial for cognitive robotics, land, marine, UAV navigation, surveying and localization applications. In these domains metric data can be utilized in reasoning when available. Additionally, qualitative or coarse spatial information is also relevant because human agents tend to use symbolic terms of natural language; moreover spatial data obtained from sensors may be imprecise. In this paper, we have proposed a new hybrid formalism $HOPRA$ for position, orientation, distance and an ASP-based framework for reasoning. This framework can solve consistency problem of *OPRA*, and incorporate distance and quantitative constraints to it; the existing literature does not have solution to these problems. We have also examplified a guidance mechanism (Section 5) for a spatial reasoning problem to improve the pruning and search process for reaching a solution faster.

References

[Baral2003] Chitta Baral. *Knowledge Representation, Reasoning, and Declarative Problem Solving*. Cambridge University Press, New York, NY, USA, 2003.

[Barták2001] Roman Barták. Theory and practice of constraint propagation. In *Proceedings of the 3rd Workshop on Constraint Programming in Decision and Control*, volume 50, 2001.

[Baryannis *et al.*2018] George Baryannis, Ilias Tachmazidis, Sotiris Batsakis, Grigoris Antoniou, Mario Alviano, Timos Sellis, and Pei-Wei Tsai. A trajectory calculus for qualitative spatial reasoning using answer set programming. *Theory and Practice of Logic Programming*, 18(3-4):355–371, 2018.

[Baryannis *et al.*2020] George Baryannis, Ilias Tachmazidis, Sotiris Batsakis, Grigoris Antoniou, Mario Alviano, and Emmanuel Papadakis. A generalised approach for encoding and reasoning with qualitative theories in answer set programming. *Theory and Practice of Logic Programming*, 20(5):687–702, 2020.

[Brenton *et al.*2016] Christopher Brenton, Wolfgang Faber, and Sotiris Batsakis. Answer set programming for qualitative spatio-temporal reasoning: Methods and experiments. In *OASICS-OpenAccess Series in Informatics*, volume 52. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2016.

[Cheng *et al.*1999] BMW Cheng, Kenneth M. F. Choi, Jimmy Ho-Man Lee, and JCK Wu. Increasing constraint propagation by redundant modeling: an experience report. *Constraints*, 4:167–192, 1999.

[Clementini *et al.*1997] Eliseo Clementini, Paolino Di Felice, and Daniel Hernández. Qualitative representation of positional information. *Artificial intelligence*, 95(2):317–356, 1997.

[Dylla and Moratz2004] Frank Dylla and Reinhard Moratz. Empirical complexity issues of practical qualitative spatial reasoning about relative position. In *Workshop on Spatial and Temporal Reasoning at ECAI*, volume 2004, 2004.

[Dylla *et al.*2007] Frank Dylla, Lutz Frommberger, Jan Oliver Wallgrün, Diedrich Wolter, Berhard Nebel, and Stefan Wölfl. Sailaway: Formalizing navigation rules. In *Proceedings of the Artificial and Ambient Intelligence Symposium on Spatial Reasoning and Communication, AISB*, volume 7, pages 1–5, 2007.

[Frommberger2008] Lutz Frommberger. Learning to behave in space: A qualitative spatial representation for robot navigation with reinforcement learning. *International Journal on Artificial Intelligence Tools*, 17(03):465–482, 2008.

[Gebser *et al.*2012] Martin Gebser, Roland Kaminski, Benjamin Kaufmann, and Torsten Schaub. *Answer Set Solving in Practice*. Synthesis Lectures on Artificial Intelligence and Machine Learning. Morgan & Claypool Publishers, 2012.

[Gelfond and Kahl2014] Michael Gelfond and Yulia Kahl. *Knowledge Representation, Reasoning, and the Design of Intelligent Agents: The Answer-Set Programming Approach*. Cambridge University Press, New York, NY, USA, 2014.

[Gelfond and Lifschitz1988] M. Gelfond and V. Lifschitz. The stable model semantics for logic programming. In *Proc. of ICLP*, pages 1070–1080. MIT Press, 1988.

[Gelfond and Lifschitz1991] Michael Gelfond and Vladimir Lifschitz. Classical negation in logic programs and disjunctive databases. *New generation computing*, 9(3-4):365–385, 1991.

[Glez-Cabrera *et al.*2013] Francisco J Glez-Cabrera, José Vicente Álvarez-Bravo, and Fernando Díaz. Qrpc: A new qualitative model for representing motion patterns. *Expert systems with applications*, 40(11):4547–4561, 2013.

[Izmirlioglu and Erdem2023] Yusuf Izmirlioglu and Esra Erdem. Qualitative reasoning about 2d cardinal directions using answer set programming. *Journal of Artificial Intelligence Research*, 77:1371–1453, 2023.

[Karwan *et al.*2012] Mark H Karwan, Vahid Lotfi, Jan Telgen, and Stanley Zionts. *Redundancy in mathematical programming: A state-of-the-art survey*, volume 206. Springer Science & Business Media, 2012.

[Lee2014] Jae Hee Lee. The complexity of reasoning with relative directions. In *ECAI 2014*, pages 507–512. IOS Press, 2014.

[Li2012] Jason Jingshi Li. Qualitative spatial and temporal reasoning with answer set programming. In *Proc. of ICTAI*, volume 1, pages 603–609. IEEE, 2012.

[Lifschitz2019] Vladimir Lifschitz. *Answer Set Programming*. Springer, 2019.

[Lücke *et al.*2011] Dominik Lücke, Till Mossakowski, and Reinhard Moratz. Streets to the opra—finding your destination with imprecise knowledge. In *Proc. IJCAI Workshop on Benchmarks and Applications of Spatial Reasoning*, pages 25–32, 2011.

[Maaref and Barret2002] Hichem Maaref and Claude Barret. Sensor-based navigation of a mobile robot in an indoor environment. *Robotics and Autonomous systems*, 38(1):1–18, 2002.

[Miguel-Tomé2021] Sergio Miguel-Tomé. The heuristic of directional qualitative semantic: A new heuristic for making decisions about spinning with qualitative reasoning. *Robotics*, 10(1):17, 2021.

[Moratz and Ragni2008] Reinhard Moratz and Marco Ragni. Qualitative spatial reasoning about relative point position. *Journal of Visual Languages & Computing*, 19(1):75–98, 2008.

[Moratz *et al.*2005] Reinhard Moratz, Frank Dylla, and Lutz Frommberger. A relative orientation algebra with adjustable granularity. In *Proceedings of the workshop on agents in real-time and dynamic environments (IJCAI 05)*, volume 21, page 22, 2005.

[Mossakowski and Moratz2010] Till Mossakowski and Reinhard Moratz. Qualitative reasoning about relative direction on adjustable levels of granularity. *arXiv preprint arXiv:1011.0098*, 2010.

[van Emden1999] Maarten H van Emden. Algorithmic power from declarative use of redundant constraints. *Constraints*, 4:363–381, 1999.

[Wallgrün2010] Jan Oliver Wallgrün. Qualitative spatial reasoning for topological map learning. *Spatial Cognition & Computation*, 10(4):207–246, 2010.

[Wolter and Lee2010] D Wolter and JH Lee. On qualitative reasoning about relative point position. *Artificial Intelligence*, 174:1498–1507, 2010.