Qualitative Spatial Reasoning for Boosting Learning-Based Robotics

Diedrich Wolter
Smart Environments
University of Bamberg, Germany
Email: diedrich.wolter@uni-bamberg.de

Alexandra Kirsch
Human-Computer Interaction
University of Tübingen, Germany
Email: alexandra.kirsch@uni-tuebingen.de

***Abstract*—Learning and motion planning are powerful methods that exhibit complementary strengths. While planning allows goal-directed actions to be computed when a reliable forward model is known, learning allows such models to be obtained autonomously. In this paper we outline how both methods can be combined using an expressive qualitative knowledge representation as a link. We argue that the crucial step in this integration is to employ a representation based on a well-defined semantics which empowers reasoning to boost performance of learning as well as of the resulting action plans. We present an architecture for learning based robotics that exploits qualitative reasoning. Expected beneficiaries of this approach are discussed, some of which are already demonstrated in proof-of-concept experiments.**

Keywords*—*qualitative spatial and temporal reasoning, task representation, learning, AI robotics

I. Introduction

Robotic applications have evolved considerably, yet they still lack the efficiency, flexibility and adaptability that is needed to improve our everyday life as versatile service robots. *Planning* and *learning* are the two major paradigms employed to make a robot act intelligently. Planning, a form of symbolic reasoning that is now applied on various levels of abstraction of state spaces, is successful whenever reliable forward models of the robot and its environment are available. Learning allows such models to be obtained. It requires carefully handcrafted state spaces to yield convergence in a reasonable amount of iterations. Neither planning nor learning on their own suffice to realize versatile service robots, but both paradigms need to be integrated, benefiting from their respective strengths.

Performing such integration also needs to pay attention to the fact that service robots are not purely autonomous robots, but they need to interact with humans – within a human society as studied in the field of social robotics, and individually with the humans they provide service for as studied in the area of human-robot interaction. Therefore, a solid approach to integrating planning and learning should not be a mere glue layer between two distinct paradigms, but rather it needs to act as a central hub that mediates between learning, planning, and interaction.

Among the various aspects of knowledge and skills a robot requires to master its applications, we are particularly interested in spatial and temporal aspects since these are elementary for any manipulation task, for example, housekeeping, serving drinks, playing board games etc. In combination with further physical knowledge, possession of spatio-temporal knowledge enables a robot to solve everyday manipulation tasks using learning and planning. In this paper we are concerned with the question of how the desired hub can be realized with respect to aspects of spatio-temporal knowledge. As a simple running example for this paper we choose throwing objects, e.g., into a dustbin (see Fig. 3 for a household scenario in a robot simulator) or learning to perform a trick shot (e.g., the ball has to bounce at least three times before hopping into the goal from behind). Like with most manipulation tasks, coarse background knowledge is available to the robot designer (naive physics, for example) that, in combination with learning and action planning, provides information to master a task.

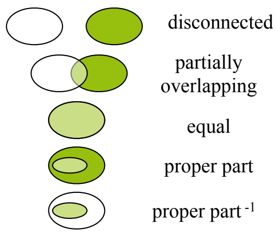
Our work is motivated by the hypothesis that qualitative spatial representation and reasoning techniques offer the means to realize the desired integration and enable us to boost learning-based robotic architectures. These provide grounding of spatial concepts and provide fundamental reasoning techniques, for example to infer new knowledge. This allows further symbolic reasoning techniques to be applied, for example planning. Technically speaking, we propose to use qualitative spatio-temporal representations to link the level of observation and control with an abstract state representation. Sinc concepts in qualitative spatial representations can reflect the cognitive level of human spatial understanding, we also obtain a basis for capturing natural language semantics for verbalization and natural language understanding [1]. In other words, qualitative representations are a promising candidate to fulfill the role of the envisaged hub. Our research aims to evaluate the potential contribution of qualitative representation and reasoning techniques. In the following we will focus on the integration of learning and planning.

The aim of this position paper is to discuss a general approach of how qualitative reasoning methods can be applied within a learning-based robot architecture to integrate learning and motion planning. One central aim of this paper is to provide the basis for discussing a research agenda to gain an in-depth understanding of the contribution of qualitative approaches. To support our point we present first results from an empirical evaluation on applying qualitative reasoning for improving performance in learning-based robotics to undermine our claims. The empirical findings presented in this paper have been previously published and discussed in detail in [2].

The remainder of this paper is structured as follows. First, Section II provides an introduction to qualitative representations in general and our spatial logic QSL for task specification in particular. Section III then discusses an architecture that

relation	term	example	definition
b	x before y	xxx	$x_e < y_s$
bi	y after x	yyy	
m	x meets y	xxxx	$x_e = y_s$
mi	y met-by x	yyyy	
o	x overlaps y	xxxxx	$x_s < y_s < x_e \wedge$
oi	y overlapped-by x	yyyyy	$x_e < y_e$
d	x during y	xxx	$x_s > y_s \wedge$
di	y includes x	yyyyyyy	$x_e < y_e$
s	x starts y	xxx	$x_s = y_s \wedge$
si	y started-by x	yyyyyyy	$x_e < y_e$
f	x finishes y	xxx	$x_e = y_e \wedge$
fi	y finished-by x	yyyyyyy	$x_s > y_s$
eq	x equals y	xxxxxxx	$x_s = y_s \wedge$
		yyyyyyy	$x_e = y_e$

(a) relations in Allen's interval algebra between intervals (x_s, x_e) and (y_s, y_e)



(b) relations in RCC-5

makes use of the qualitative representation to link learning with motion planning and exploits reasoning to boost performance of. Section IV summarizes first results. Finally, we sum up and conclude.

II. What is qualitative spatial and temporal reasoning?

Qualitative representations employ symbols to represent semantically meaningful concepts of an underlying domain, they abstract from quantitative details irrelevant for a task at hand. This creates a task specificity leading to a high variety of qualitative representations developed so far (for an overview, see, e.g., [3]). Moreover, spatial and temporal domains are typically continuous and exhibit complex structures that can be exploited in several ways. Algorithmically speaking, reasoning tasks can benefit from specifics of a qualitative representation. Among the individual aims motivating research in qualitative representation, the overall goal of developing efficient and effective reasoning algorithms that enable the design of intelligent systems stands out. Designing a qualitative representation starts with two design decisions:

1) to identify a finite set of concepts
2) to fix a knowledge representation (language) to compose statements

Spatial and temporal concepts meaningful for a class of tasks are typically relative since there are no universally 'important' values. Concepts are thus mostly relations which are also called *qualitative relations*. Among these, binary relations have received most attention. The most prominent representatives in the collections of qualitative relations are, on the temporal side, Allen's Interval Algebra (IA) [4] which identifies 13 interval relationships (see Fig. 1(a)) and, on the spatial side, the Region Connection Calculi (RCC) which identifies topological relations between regions (see Fig. 1(b)). As a minimal qualitative representation, the set of ordering relations $\{<, =, >\}$ has been considered and, in combination with some symbolic reasoning techniques, it is referred to as the point calculus [5].

The most simple qualitative knowledge representation considered is the conjunctive composition of relational statements like $\{\text{IN}(\text{can}, \text{dustbin}), \text{ON}(\text{dustbin}, \text{ground}), \ldots\}$, also called constraints – *constraint-based reasoning* is hence an elementary form of qualitative reasoning that has been intensively studied. Due to the complex interdependencies occurring in qualitative spatial relations, defining relations and how they can be composed to form statements is only the first step in designing a qualitative representation. The difficult step is then to identify reasoning algorithms that can process these relations, most importantly to identify relations only given implicitly and to recognize potential conflicts [6].

A. Spatial logics

Manipulation tasks are inherently dynamic and thus a spatial representation suitable for task specification needs to be augmented to capture change over time. For example, we dare to represent a learning task declaratively by saying that a start configuration (e.g., $\text{ON}(\text{can}, \text{desk})$) should be changed to obtain some desired end configuration (e.g., $\text{IN}(\text{can}, \text{dustbin})$. Sequences of such statements are also called snapshot-based spatio-temporal models [7]. For representation of manipulation action, sequences of spatial relations have been considered (see [8]), but these are not suited to represent and reason about task specifications since they prescribe action sequences. By contrast, in a task specification we wish to say that a start configuration shall be transformed to *eventually* obtain a desired goal configuration and it is up to learning and motion planning to figure out how this can be accomplished. In its generality, the question of combining a qualitative spatial representation with a temporal representation is tackled in the contemporary field of *spatial logics* [7].

B. Specifying manipulation tasks with QSL

We propose to approach representing the development of qualitative spatial knowledge over time using linear temporal logic (LTL) [9]. LTL extends propositional logic by interpreting propositional with respect to an infinite linear sequence of assignments, called worlds. A statement may hold in one world, and it may be false in another. The logic can easily be adapted to accommodate qualitative constraints. LTL has previously been applied in robotics for various tasks, including motion planning [10], [11].

In short, we use plain LTL but replace all propositional symbols with qualitative constraints. As a result we obtain a temporal logic whose primitives are qualitative spatial relations, we thus have termed the logic *qualitative spatial logic* (QSL) [2]. QSL is similar to LTL in many regards (a comprehensive introduction to LTL may be found in [12]). A well-formed formula ϕ in QSL is defined by the following grammar rule:

$$\phi := C \mid \neg\phi \mid \phi_1 \wedge \phi_2 \mid \circ\,\phi \mid \phi_1 U \phi_2 \qquad (1)$$

Here, C denotes a qualitative constraint in the form $r(X, Y)$ where r is a qualitative relation and X, Y are variables that range over the spatial domain. For brevity, we only define a minimal set of operations here. Other Boolean operations can easily be expressed by term rewriting, e.g., $\phi \vee \psi :\Leftrightarrow \neg(\neg\phi \wedge \neg\psi)$. Semantics of LTL can be defined using a Kripke structure $\langle \mathbb{N}, I \rangle$: The interpretation function I assigns truth values to the constraints independently for all time points $t \in \mathbb{N}$. Technically, $I : C \to 2^{\mathbb{N}}$ maps a constraint to the set of time points at which it is satisfied.

As an example, consider the two snapshots: at time $t = 0$, the constraint $\text{TOUCHES}(\text{b}, \text{g})$ is not satisfied, at time $t = 1$ it is. A Kripke model representing this development of the world would interpret TOUCHES(b, g) to be false at time $t = 0$, so we have $0 \notin I(\text{TOUCHES}(b, g))$, but $1 \in I(\text{TOUCHES}(b, g))$. If we imagine the ball to be bouncing off three times, we would have $I(\text{TOUCHES}(b, g)) = \{1, 4, 6, 8, 9, 10, \ldots\}$. While the propositional conjuncts relate statements within one world (i.e., one snapshot in time), the operators $\circ$ (next) and U (until) relate statements across worlds. They allow us to state that, for example, in the text point in time $\text{TOUCHES}(b, g)$ will hold by writing $\circ\text{TOUCHES}(b, g)$ or that a ball will always be above ground before it touches ground by writing $\text{ABOVE}(b, g)\ U\ \text{TOUCHES}(b, g)$.

Given an interpretation $I(C_k)$ of all spatial constraints C_k for all worlds, the task of deciding whether a sub-sequence of the world is a satisfying assignment of a given formula is called *model checking*. We apply model checking for instance to identify whether an observed process suits a declarative description of the task to be performed. The interpretation of the constraints is thus determined by the observation. A constraint is $r(X, Y)$ is said to be true, if the relation r between the objects referred to by variables X and Y is observed.

Beyond knowing that a sequence of observations constitutes a model for a formula, we are also interested in identifying *when* the formula gets true. For example, by model checking the formula $\text{ABOVE}(\text{ball}, \text{ground}) \wedge \circ\text{ON}(\text{ball}, \text{ground})$ we wish to identify the first time point at which ball and ground get into contact. To this end, we adapt the notion of model checking to be task of computing the first time point a formula gets true, returning ∞ if a sequence of observations provides no model. Based on this understanding of model checking QSL also provides the basis for a language to represent useful background knowledge, for example to say that if the first contact with ground is BEHIND a goal that shall be hit with the ball, then one has thrown too far.

III. QUALITATIVE SPATIAL AND TEMPORAL REASONING FOR LEARNING-BASED ROBOT ARCHITECTURES

In the following we first describe how we approach learning and planning in manipulation task based on qualitative representation and reasoning. From this discussion we derive claims of how qualitative reasoning improves performance by better linking learning and planning.

To achieve the goal of a manipulation task, a robot has to determine a sequence of actuator control commands. Learning such a sequence directly would require a temporal component, which would result in a huge state space that no learning algorithm to date can handle, not to mention the enormous amount of data that would be required. We propose a different approach of learning independent forward models: how a motor command affects the robot state, whether a joint configuration would cause a collision, and whether a joint configuration securely grasps a desired target object. These forward models are then used with planning to determine the desired action sequence by alternately creating an abstract plan (i.e., a model of the spatio-temporal logic formula that describes the manipulation task) and verifying its feasibility by the motion planner to connect the consecutive qualitative configurations. The execution of the final action sequence is continually monitored for collisions and the achievement of the manipulation goal. Figure 1 illustrates our overall approach, highlighting the components based on qualitative representations and four ways of how qualitative reasoning can serve to improve learning-based robotics. Let us explain the architecture at whole by describing how a learning task is mastered.

1) Integrating learning and programming: The entry point is a declarative task specification that is part of the robot control program that shall automatically be turned into appropriate robot control commands by means of learning and planning. This integration has for example been pursued by Thrun [13] who proposed the language CES offering the possibility to leave "gaps" in the code that can be closed by learned functions. CES requires the programmer to provide suitable training examples as experience acquisition is not supported on the language level [14]. There exist approaches similar to CES, each focus on different aspects of how learning can be applied to automatically complete a robot control program. We propose to employ the Robot Learning Language (RoLL) [15] since it already offers language constructs to specify learning problems and experiences in a declarative way, easing the integration with QSL. RoLL is available as an open-source ROS package[1].

2) State representation for robot learning and problem decomposition: Machine learning crucially depends on an adequate state space representation. Different approaches are pursued to find a suitable abstraction that yields the desired representation. Some branches of machine learning, such as Deep Learning [16] try to integrate the state abstraction into the fully automated learning process, relieving the programmer from explicitly specifying an abstract state space. However, the automatically generated state spaces are black boxes and it is unclear how we can reason about a state space if its semantics are unknown. We use qualitative spatial and temporal representations to represent the state space in a compact way, deriving the state space representation from the qualitative concepts employed in the task specification. Several methods have been proposed to automatically detect *explicit* abstract state descriptions from a vector of observation variables. For example, Stulp et al. [17] search for abstract features by applying several combination methods on the observation variables. Following these ideas it would be possible to adapt a state space representation by rewriting of the concepts employed in the task specification. This is possible due to the known semantics of the respective qualitative representations.

3) Action planning and failure analysis: Today's hybrid planning systems already integrate AI planning with robot motion planning to master challenging motion planning problems [18, e.g.,]. We propose to adapt existing methods to generate qualitative plans from qualitative spatial logic formulas that specify action goals. The utility of qualitative plans to provide intermediate goals for an action planner have already been demonstrated [19]. At this point, we require the availability of some (initial) forward models. These can be learnt from some random arm movements like a kind of motor babbling. For more complex learning tasks such as learning a physical model for the manipulated objects, generating informative training data requires sampling within a highly constrained space. Given a declarative task specification, the problem generator can automatically compute such relevant start positions by

[1] http://wiki.ros.org/roll

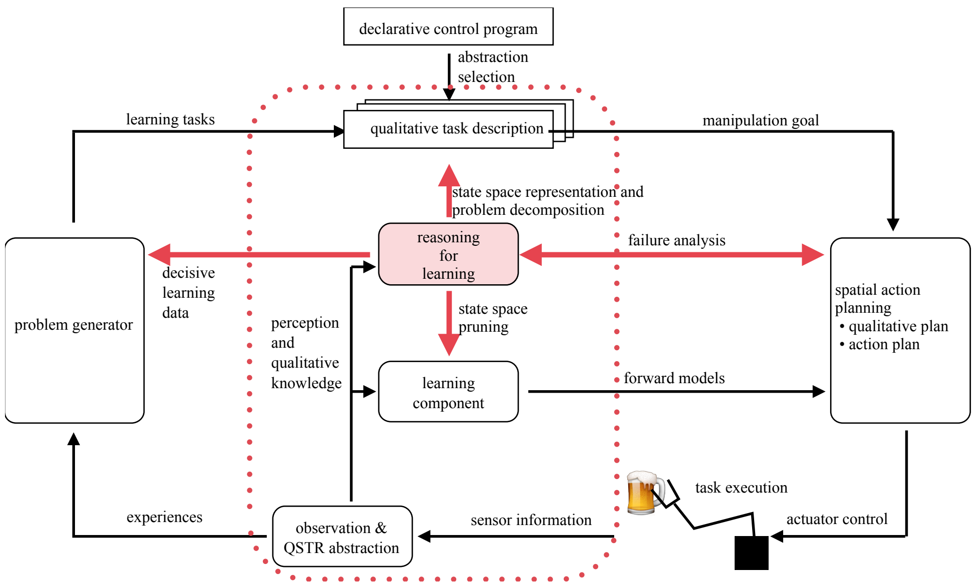


Fig. 1. Integrated learning, planning and adaptation process based on qualitative representations (components in the framed central part) with qualitative reasoning (highlighted)

computing models for the start configurations specified using qualitative reasoning techniques.

On an adequate level of qualitative abstraction motion planning can then be performed in a step-wise fashion connecting one intermediate goal to the next by sampling-based methods using the learned forward models (see [20]). Sampling-based approaches to planning are attractive since they can easily be combined with qualitative reasoning to perform failure analysis in context of background knowledge. For example, some laws of naive physics can be modeled in QSL and model checked against sampled plans that fail to reach the desired goal. If a law matches the sampled plans it provides an explanation why the plan failed and hence how it can be improved, i.e., where in the state space a solution is more likely to be found. For example, consider the ball throwing task shown in Fig. 2 in which a ball is to be thrown to land in a goal area. Here, the robot can infer that it is pointless throwing the ball to the left since it cannot bounce off any obstacle, changing direction towards the goal. Moreover, if the robot tries to hit the goal using some value φ as launch angle and some arm speed v it can observe the trajectory and reason about failures. From Fig. 2 trajectory 2, for example, the robot can infer that using a flatter angle with less force will not work out either; after just 2 trials the configuration space necessary to explore can be pruned significantly as marked red in Fig. 2 left.

4) Pruning and partitioning the state space: The qualitative concepts used in a task specification allow the state space to be partitioned into qualitatively distinct sub-problems. For example, in context of the ball throwing a task specified using concepts like left/right and above/below the partitioning shown in Fig. 2 leads to several distinct sub-problems, depending on where the robot is positioned with respect to obstacles and goal. Learning a forward function specifically for a sub-problem can ease learning of complex functions, in particular if forward functions are complex. By model checking a concrete instance of the task at hand, the robot can automatically decide which of the sub-problem it is facing and hence choose the right

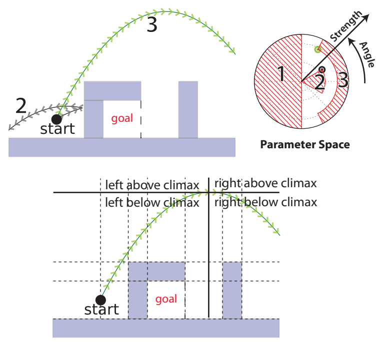


Fig. 2. Top: Illustration of qualitative reasoning for state space pruning and problem generation in the domain of throwing a ball. Bottom: example qualitative partitioning with selected labels.

forward model for planning. The approach to failure analysis can also be applied to prune the state space used in the learning component. Once a part of the search space is known to not be relevant for solving a learning task, these parts of the state space representations can be pruned off.

5) Problem generator and decisive learning problems: Efficiency of learning is crucial for acceptability of service robots. Robots requiring several hundred attempts to learn new tasks are not acceptable, let alone the side effects of how task execution could go wrong. The reason for the large number of training data required is that large and heterogeneous state spaces need to be covered. The qualitative partitioning allows training experiences to be evenly distributed with respect to the qualitatively different configurations. This yields a task-sensitive problem generation and thus requires fewer instances to be generated than by uninformed sampling. For example, it is less useful to generate many training instances for the aforementioned ball throwing task with start and goal located in direct line of sight (convex enclosure of start and goal not overlapping with obstacles), but few instances only for the

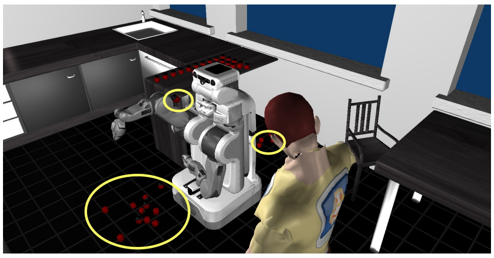


Fig. 3. Result from a data gathering run, showing highlighted landing positions on the floor behind the robot, on the robot, and the desired samples on the floor in front of the robot.

different combinations of how obstacles can obstruct a direct line of flight.

To sum up, we see several potential benefits of linking learning and planning by means of a qualitative representation that also supports reasoning:

- Claim 1: better learning performance through problem decomposition
- Claim 2: better planning performance through failure analysis
- Claim 3: improved learning speed through state space pruning
- Claim 4: improved learning performance through adaptive learning data generation

IV. Experiments

We are currently involved with investigating to which extent our claims are satisfied. For some of the claims, we can already give first results that demonstrate the utility of the reasoning-based architecture we propose. For an in-depth description of the experiments summarized in the following, please consult [2]. In our experiments we simulate a Willow Garage PR2 robot using the simulator MORSE[2] [21]. Since neither the robot nor the middleware ROS have been designed for throwing objects, implementing the ball throwing scenario already turns out to be a challenging learning task. See Fig. 3 for a screen shot of the simulation after the robot has thrown several objects. To make the robot throw an object (in our case an empty jar), we first move the elbow and wrist joints of the right arm to a low position. Then the robot moves the two joints to a new position, with a specified time to perform the movement. After half the time that the movement should last, the robot lets go of the object. Although the arm movement was controlled such that the objects should have flown to the front of the robot, several objects bounced off the robot (e.g., if there was a double contact when releasing the object), resulting in the wide range of landing positions shown in Fig. 3.

Claim 1: better learning performance through problem decomposition

We collected 158 training instances by commanding the robot to throw an object to the front. These include runs, where the object landed behind or on top of the robot. By measuring the distance and angle with respect to the robot's front achieved

TABLE I. Learning performance for forward models. MAE: mean absolute error, RMSE: root mean squared error

filter	number of training instances	learning errors direction MAE	learning errors direction RMSE	learning errors distance MAE	learning errors distance RMSE
no	158	0.8058	1.431	0.2092	0.2565
yes	80	0.0103	0.0178	0.0939	0.136

we train two forward models using M5' algorithm from the WEKA toolbox [22] based on the parameters of the throwing action (in our case five variables: elbow joint at start/end configuration, wrist joint at start/end configuration, and time in which the action is to be performed). We compared the results of the learning element applied to the whole set of training instances with results after applying problem decomposition. To this end, we specify a throw to the front using QSL saying that the first contact to ground needs to be in front of the robot. By model checking the training instances against the specification we can filter out those that do not represent a throw to the front. The results in terms of mean absolute error and root mean squared error as determined by WEKA are shown in Table I. Despite reduced training instances the learning performance is significantly improved. The mean absolute learning error for the distance model by -55.1% (-47.0% for the root mean squared error) and for the direction model by -98.7% (-98.8% for the root mean squared error). These results suggest that problem decomposition can offer very effective means for improving learning performance.

Claim 2: better planning performance through failure analysis

In context of the ball throwing task, coarse background knowledge for failure analysis is provided by naive physics which can easily be provided by the programmer. We investigated how sampling-based planning can be improved by reasoning about samples not leading to the goal state. For this experiment, we isolated the physical simulation underlying MORSE which is based on the Bullet library[3]. As parameter for throwing we consider horizontal and vertical velocities (v_h and v_v, respectively) and we perform simulation with added noise. Let d stand for the distance achieved for a specific choice of v_h, v_v and let g be the desired target distance. Rules to identify a choice of parameters that is more likely to achieve the desired distance can be written down as follows in the form premises $\rightsquigarrow$ conclusion:

$$d > g \quad \rightsquigarrow v_h^+ < v_h \wedge v_v^+ < v_v \tag{2}$$

$$d < g \quad \rightsquigarrow v_h^+ > v_h \wedge v_v^+ > v_v \tag{3}$$

Note that the left and right hand side of $\rightsquigarrow$ are expressible with QSL using the point calculus to relate the variables. Variables v_h+ and v_v^+ stand for suitable values of future action parameters that should be considered in sampling. Note further that the rules oversimplify as they demand to change both parameters at the same time although alerting either one affects the distance achieved. We apply these rules by updating a probability distribution of the parameter space of v_h, v_v which is used for sampling. Figure 4 illustrates this effect for the task constrained to $v_h = v_v$ which leads to a one-dimensional distribution better suited for visualization: after few samples, the distribution concentrates around the desired action parameter.

[2]https://www.openrobots.org

[3]bulletphysics.org

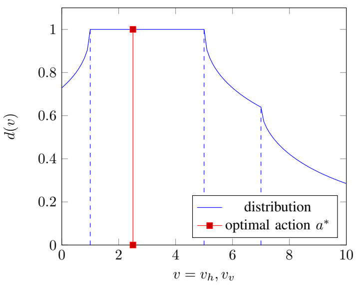


Fig. 4. Updating the representation of a distribution based on sampled actions and qualitative inference leads to a concentration around the desired but unknown action a^* after evaluating actions $a = 5$,1, and 7.

TABLE II. PLANNING QUALITY MEASURED IN DEVIATION FROM GOAL

failure analysis	avg. deviation from goal [m]	std. mean error
no	0.028	0.083
yes	0.011	0.040

To reveal the effect in practice, we ran 100 trials to make the robot throw a ball a randomly selected goal distance. During planning, 50 samples are drawn (which proved to be sufficient in this simple task). The results shown in Tab. II indicate that the QSL-based method exploiting qualitative reasoning for failure analysis lowers the error on average, but also reduces the spread of the results. The difference is statistically significant ($p = 0.0001$). Due to using a qualitative representation with clear semantics, applying qualitative reasoning is straightforward and helps to improve quality of action plans generated. Still, in all cases, the computed actions are predicted to be very close to the true distance.

V. Summary and Conclusion

In this paper we propose a reasoning-based architecture to combine learning and planning using a qualitative spatial representation as hub. As scenario we consider service robots that need to accomplish manipulation tasks that require a careful consideration of (initially unknown) physical parameters such as friction, mass, etc. In the area of qualitative spatial and temporal reasoning, several representations have been developed wich can be applied in context of robot manipulation tasks to specify tasks and to capture laws of naive physics. Qualitative spatial representations offer a well-defined semantics that, unlike arbitrary symbols obtained by learning, enable application of symbolic reasoning. We argue that the ability to reason is crucial to boost the performance of learning and planning and derive four claims of how reasoning allows performance of a robotic system to be improved. We give first results that indicate the utility of the overall approach. Implementing the proposed architecture within the robot learning language RoLL and investigating all claims in detail is subject to future work.

Moreover, in a fully automated integration of learning and planning into a robot control language, the language needs to provide a runtime system capable of understanding reasons of failure in order to provide a cure. This is possibly the most challenging endeavor in the envisaged integration of learning and planning and we are interested to investigate whether reasoning can contribute to these challenges too.

References

[1] R. Moratz and T. Tenbrink, "Spatial reference in linguistic human-robot interaction: Iterative, empirically supported development of a model of projective relations," *Spatial Cognition & Computation*, vol. 6, no. 1, pp. 63–107, 2006.

[2] D. Wolter and A. Kirsch, "Leveraging qualitative reasoning to learning manipulation tasks," *robotics*, vol. 4, no. 3, pp. 253–283, 2015, special issue representations and reasoning for robotics.

[3] J. Chen, A. G. Cohn, D. Liu, S. Wang, J. Ouyang, and Q. Yu, "A survey of qualitative spatial representations," *Knowledge Engineering Review*, vol. 30, pp. 106–136, 2013.

[4] J. F. Allen, "Maintaining knowledge about temporal intervals," *Communications of the ACM*, vol. 26, no. 11, pp. 832–843, Nov. 1983.

[5] M. B. Vilain and H. A. Kautz, "Constraint propagation algorithms for temporal reasoning," in *Proceedings of AAAI*, 1986, pp. 377–382.

[6] J. Renz and B. Nebel, "Qualitative spatial reasoning using constraint calculi," in *Handbook of Spatial Logics*, M. Aiello, I. E. Pratt-Hartmann, and J. F. van Benthem, Eds. Springer, 2007, pp. 161–215.

[7] R. Kontchakov, A. Kurucz, F. Wolter, and M. Zakharyaschev, "Spatial logic + temporal logic = ?" in *Handbook of Spatial Logics*, M. Aiello, I. E. Pratt-Hartmann, and J. F. van Benthem, Eds. Springer, 2007, pp. 497–564.

[8] E. E. Aksoy, M. Tamosiunaite, and F. Wörgötter, "Model-free incremental learning of the semantics of manipulation actions," *Robotics and Autonomous Systems*, vol. 71.

[9] A. Pnueli, "The temporal logic of programs," in *Proceedings of the 18th Annual Symposium on Foundations of Computer Science (FOCS)*, 1977, pp. 46–57.

[10] S. L. Smith, J. Tůmová, C. Belta, and D. Rus, "Optimal path planning under temporal logic constraints," in *Proceeding of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Taipeh, Taiwan, October 2010, pp. 3288–3293.

[11] M. Kloetzer and C. Belta, "Automatic deployment of distributed teams of robots from temporal logic motion specifications," *IEEE Transactions on Robotics*, vol. 26, no. 1, pp. 48–61, 2010.

[12] F. Kröger and S. Merz, *Temporal Logic and State Systems*, ser. Texts in Theoretical Computer Science. Springer, 2008.

[13] S. Thrun, "Towards programming tools for robots that integrate probabilistic computation and learning," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. San Francisco, CA: IEEE, 2000.

[14] S. Thrun, "A framework for programming embedded systems: Initial design and results," Carnegie Mellon University, Computer Science Department, Pittsburgh, PA, Tech. Rep. CMU-CS-98-142, 1998.

[15] A. Kirsch, "Integration of programming and learning in a control language for autonomous robots performing everyday activities," Ph.D. dissertation, Technische Universität München, 2008. [Online]. Available: http://mediatum2.ub.tum.de/node?id=625553

[16] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 8, pp. 1798–1828, 2013.

[17] F. Stulp, M. Pflüger, and M. Beetz, "Feature space generation using equation discovery," in *Proceedings of the 29th German Conference on Artificial Intelligence (KI)*, 2006.

[18] J. Wolfe, B. Marthi, and S. Russell, "Combined task and motion planning for mobile manipulation," in *Proceedings of ICAPS*, 2010.

[19] M. Westphal, C. Dornhege, S. Wölfl, M. Gissler, and B. Nebel, "Guiding the generation of manipulation plans by qualitative spatial reasoning," *Spatial Cognition and Computation: An Interdisciplinary Journal*, vol. 11, no. 1, pp. 75–102, 2011.

[20] L. Mösenlechner and M. Beetz, "Parameterizing Actions to have the Appropriate Effects," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, San Francisco, CA, USA, September 25–30 2011.

[21] G. Echeverria, N. Lassabe, A. Degroote, and S. Lemaignan, "Modular openrobots simulation engine: MORSE," in *Proceedings of the IEEE ICRA*, 2011.

[22] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten, "The weka data mining software: An update," *SIGKDD Explorations*, vol. 11, no. 1, 2009.