Contextualization through Simulation

John A. BATEMAN, Mihai POMARLAN and Gayane KAZHOYAN
Bremen University, Bremen, Germany

Abstract. It is well known that the semantics of human utterances only partially covers the meaning the speaker wishes to convey to the listener, because the listener can be expected to fill in gaps, specialize descriptions, and resolve ambiguities by using context clues. Context is often broadly construed to include previous utterances, but may also contain information about the present situation or history of the communicators, such as what task apart from communication they are trying to do now or about to engage in. This underspecificity of human language is a problem for interactions between humans and service robots, because, unlike a human listener, a robot presently cannot be expected to ably employ contextual clues to fill in semantic gaps. Further, because the nature of the context clues is extremely varied, one can expect the reasoning mechanisms that would be able to handle this challenge to be very heterogeneous also. In this paper, we present a framework to tackle aspects of contextual inference related to the interpretation of spatial relations by combining ontological engineering principles and situated embodied simulations of robotic agents. These simulations are a new reasoning tool to find constraints on human command interpretation.

Keywords. embodiment, simulation, linguistic semantics, ontological analysis, formal ontology

1. Introduction: background and motivations

The relation between *descriptions*, the bearers of meaning exchanged in communication, and the contexts which those descriptions are taken to pick out, is a complex and flexible one. It goes further than just adding contextual information on top of what the descriptions contain (the classic semantics–pragmatics divide [20]). Descriptions suggest or constrain contexts, which makes it challenging to relate levels of description, e.g. linguistic utterances, with actual contexts of use, e.g. a physical environment in which language users are embedded in.

Our proposed framework approaches this issue through a combination of ontological engineering principles and situated embodied simulations involving robotic agents. Simulation allows implementation and experimentation at a level of abstraction not usually accessible to linguistic analysis alone. Also, it increasingly appears to be the case that some form of simulation may play a crucial role in language comprehension and production for humans as well.

We will focus on the contextualization issues for the semantics of natural language construals of space, spatial entities, events and activities unfolding in space and time [7]. Contextualization is seen as embodied simulations in which actions and movements can be performed corresponding to the linguistic descriptions offered. Actions and movements will vary drawing on a range of parameters, thereby providing a formalized connection between linguistic descriptions and contextualized representation and considerable flexibility concerning just what actions and movements ensue.

Our problem setting is an application scenario drawn from service robotics, which showcases the issues of contextualization. A robot needs to know in great detail what it is tasked to perform, or it will not do anything. Natural language instructions on the other hand are very underspecified, requiring inference on the robot's part to fill in the gaps and get to some actionable description of behavior. We demonstrate how access to simulation via appropriately managed intermediate levels of qualitative description makes relevant information derived from simulation available for the system as a whole.

2. Problem Setting: Everyday activities and their descriptions

Consider this scenario: one or several intelligent robots are performing everyday household activities with guidance or instruction from humans using natural language. Human utterances need to be somehow translated to robotic actions and movements. However, natural language describing everyday activities is often highly schematic, multiply ambiguous, and underspecified. At the same time, a robot needs very precise descriptions of the actions it is tasked to perform. This same difficulty must underlie language understanding in humans as well, however, there the necessary inference mechanisms are opaque. The robotic scenario allows us access to all aspects of cognition and its interaction with the world, including motor control, perception and sensory feedback.

A cognitive ability of our robotic system which we believe important is *simulation*: the ability to try out future actions and assess potential outcomes without performing those actions. This is likely an ability of human language comprehenders as well, and there is substantial research that suggests language use may involve partial simulations [2,17,28,13,1], even if the precise mechanisms are still unclear.

Consider then this instruction:

Put the plates on the table.

Apart from *grounding* [29] discourse elements into context, interpretation also needs to select a location to move the plates to. The sentence to interpret merely constrains this location to be in a certain relation (indicated by 'on') to a relatum (the 'table'). The manner in which 'putting' is to be performed, that is, the exact motion and resulting state of the moved objects, also needs to be selected.

We use a 'linguistically-motivated' ontology that provides an ontology-like level of semantic description for natural language called the Generalized Upper Model (GUM) [7], in which categories and relationships are motivated by *grammaticalization* patterns within natural language [4]. Such a linguistic semantics alone, however, is insufficient for the contextualization task, even if it does constrain what would be valid states of the world prior, during, and following the action. These constraints will parameterize the range of simulations employed for contextualization. What a two-level semantics, particularly the GUM treatment of space, movement and action [7], buys us here is a level of semantic representation that is 'rigid' in its characterization of discourse entities but also still supportive of flexible contextualization where quite diverse properties, aspects, and 'conceptualizations' of states of affairs can occur.

The shallow linguistic semantics of our example sentence has several components derived compositionally during analysis. The utterance as a whole evokes a generalized configuration of 'putting' or 'placing'. In standard Manchester description logic notation:

(1) $\mathsf{AffectingAction} \equiv \mathsf{Configuration} \sqcap \exists \mathsf{actee.SimpleThing} \sqcap \exists \mathsf{placement.GeneralizedLocation}$

The kind of entity that GUM allows to fill the placement role is a GeneralizedLocation:

(2) $\mathsf{GeneralizedLocation} \equiv \mathsf{Circumstance} \sqcap \exists \mathsf{hasSpatialModality.SpatialModality} \sqcap \exists \mathsf{relatum.SimpleThing}$

GUM includes a subontology of spatial relations, defined as subtypes of SpatialModality and often characterized in terms of *functional* requirements [11]. In the above example, use of 'on' invokes a *functional support* SpatialModality. Notions defining 'on' in terms of geometric properties will often not be satisfied by actual uses of the preposition (e.g., 'the painting is on the wall') and so the functional characterization offers a more generally valid description [7,3].

Here then is the semantics of the example utterance, expressed in standard SPL-notation [16]:

```
(3) (e / AffectingAction
        :actee (p / plate)
        :placement (l / GeneralizedLocation
                      :hasSpatialModality Support
                      :relatum (t / table)))
```

Characterization as functional support covers most cases where 'on' invokes a spatial location, but it does not specify an actual pose. However, contextualization may require the receiver of the utterance to form an idea of what poses are appropriate or not. The same issue appears for all spatial relationship between objects invoked by language ("in", "over", or "near"), when these are to be produced in the real world.

To decide whether a particular pose is appropriate for instantiating a functional relation, the intended *behavior* for the situation must be considered. Such behavioral characterizations lend themselves well to simulation. Therefore, to go from qualitative linguistic descriptions to actual coordinates in space can be achieved by having the linguistic part parameterize a process of sampling and validation of candidate poses.

However, the nature of the behavior described is itself also underspecified. One way to interpret the example command is that the plates should be placed such that the bottom surface of each plate contacts the top surface of the table directly. *Stacking* the plates on the table is another plausible interpretation. One has to consider then the *intended purpose* of the described action: why are the plates brought to the table? If people are about to eat from the plates, then leaving the plates stacked is inappropriate. If the plates are simply brought to a place for easy access so they can be washed, stacking may in fact be desired.

Using *ad hoc*, hard-coded 'if-then' rules for such decisions (e.g., 'if setting the table, do not stack') will not scale. Even a semi-structured environment like the home has too much variability, and such rules contribute no understanding for the robot of the purpose of its activities. What is needed, then, is a capability to answer questions like "would it be appropriate to use stacking now? if not, why not?" Without answers to such questions, a robot will be unable to achieve flexible, human-level contextualizations of even the simplest instructions. Simulation is a powerful means of acquiring such knowledge.

Contextualization problems are also highlighted in the following examples:

(4) a. place the cups/chairs near to each other
b. place the plates near the sink, I'll wash them later
c. don't place the plates too near the table edge (because we will eat from them)
d. when setting a table for eating, place the fork to the left of a plate and a knife to the right
e. put the plates and cups in the cupboard

In this case the difficulty stems from 'what is near?'. A nearness criterion will vary with the objects considered or rather with the activities the objects will participate in.

Finally, the performance of the action is also an issue of contextualization. Some items form stable stacks (like plates) and others do not (like cups). When filling up a shelf, it is better to push the first items placed there to the back, so as not to block access for more items. These constraints are often not linguistically expressed, but rather learned, in the case of humans, through embodied experience. For a robot, simulation offers a way to account for them during behavior generation.

Another action that can be performed in many ways is simple picking up. Consider:

(5) a. pick up the spoon
b. pick up the spoon from the table and use it to scoop the soup from the plate
c. you can use the spoon from the table to eat the soup" (picking is implied here)
d. pick up the spoon then crack the egg (a different grasp may be needed here than in the soup case)

There are many ways to grasp an object; Figure 1 shows some examples drawing from the robotics domain. Deciding which is appropriate often depends on the context in which it is given, in particular with respect to the task for which the picked up object will be used.

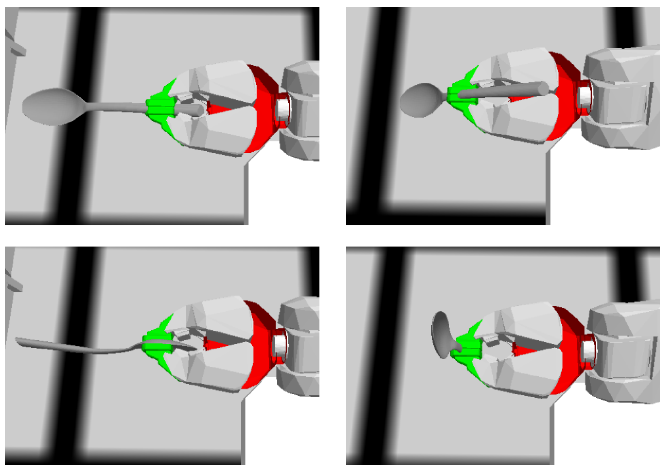

Figure 1. Some different ways to grasp a spoon. Deciding which one is appropriate will depend on context, such as what the spoon is to be used for.

Even apparently 'simple' actions may, therefore, need to be contextualized very differently – and this flexibility is precisely that targeted by our approach to combining linguistic specification with embodied simulation as we shall now see.

3. Simulation as a method to test interpretations

In the scenarios we analyze here, the robot is given a task which involves establishing, using, or destroying a spatial relation between objects, and it is unclear from the linguistic semantics how this relation is, or is to be instantiated in the world. The robot is however not acting alone – the task it performs happens in a larger context of other agents doing tasks dependent on what the robot does.

In this section, we will describe how we organize contextual information into an object we call the "execution context", and we show how it is used in parameterizing and interpreting a simulation. We will first present the robot executive program, and the process by which the semantics of a linguistic command is converted into an underspecified executable program.

3.1. The robot executive

We use the Cognitive Robot Abstract Machine [8] to generate and control the behavior of the simulated robot. CRAM is a set of tools for developing reactive programs, and includes a library of basic robot actions, logging of task executions [33], and a light-weight simulation environment called "projection" [21]. While projection abstracts away from some aspects of execution, such as arm trajectories between waypoints, it nonetheless captures interesting constraints about what configurations are reachable, which locations would result in object collisions, what are occlusion and visibility constraints, etc.

An important notion in CRAM is the designator, a feature structure that symbolically describes an object, location, or action. A symbolic description is one that does not necessarily commit to a particular object identifier or set of coordinates; for example, a designator may refer to

(6) (a location (near (an object (type cup))))

and make no mention of an actual position, or any region boundary, for points satisfying this description. However, to be actionable, a designator needs to be "grounded", that is, such specific selection information needs to be added.

This is achieved through various techniques implemented in CRAM. For locations, a set of candidates is randomly sampled from "costmaps" [34,27] then tested against a battery of validators such as absence of collision and whatever conditions may be added via the designator description, e.g. visibility or reachability. The costmaps are probability density functions initialized based on the designator content; for example, when looking for a location near an object then the costmap would be a gaussian with a maximum at the object center. The set of active samplers and validators can be changed at runtime. Programs in CRAM use designators to pass arguments to each other, and these designators are only grounded when needed.

3.2. Converting an SPL to a CRAM program

The actions in the CRAM library do not align with linguistically-motivated action categories from GUM; purely linguistic categories are too broad to be directly employed on the execution side. In this subsection we set out a translation mechanism by which underspecified linguistic semantic representations of the kind introduced above can be converted to CRAM programs for driving simulations. The characterization given here draws on the overall mechanism introduced in [25].

The conversion process applies transformation rules to semantic specifications, or semspecs, which are SPL logical forms, gradually replacing GUM configurations with function objects compositionally constructed from building blocks taken from the CRAM action library. The transformation rules have an antecedent (a pattern to look for), a consequent (a pattern to replace the antecedent with), and scope restriction on where to look for the antecedent.

We will use as a starting semspec the SPL above for the plate placement instruction (3). The first step is to obtain a CRAM designator for the actee via the following rule:

```
(7) ( (_ / ?x)
      (an object (type ?x))
      (:actee))
```

where $(\_/?x)$ is the antecedent, ?x is a variable, and the underscore ('_') unifies with anything. The consequent creates a CRAM designator for "an object of type ?x". Because of the scope restriction, the antecedent will only be matched against the value of an :actee property in the main clause of the semspec. Designators for placement locations are obtained via rules such as:

```
(8) ( (?l / GeneralizedLocation :hasSpatialModality Support
                                :relatum (_ / ?x))
      (?l / (a location (on (an object (type ?x)))))
      (:placement))
```

For an affecting action with actee and placement, our rules infer a transport action:

```
(9) ( (_ / AffectingAction :actee ?a :placement ?l)
      (Fn [(lambda () (perform (an action) (type transporting)
                                           (object ?a)
                                           (target ?l)))])
      nil)
```

where the nil scope restriction means the antecedent may match the top level of the semspec. The consequent is a function object containing an executable CRAM action.

Note that this places no constraint on the designator grounding processes. At this point, both "individual plates placed on table" and "stacked plates on table" are considered plausible interpretations for the "on" relation.

3.3. The table-setting scenarios

We will now illustrate how the ambiguity we noticed above about what "on" should mean may be resolved by simulation.

The simulation scenario is described by an "execution context", which is an object containing information about the main task and the agents and objects that participate in it, and the context it is performed in– in particular, the task(s) it is supposed to enable.

The enabled task is to be run after the main task, and may be performed by a different set of agents. In one of our running examples, the main task is a table setting task to be performed by one robot, and the enabled task is eating, to be performed by two other agents (humans, but we use simulated robots to model them as well). Table 1 shows the contents of this execution context.

Table 1. Execution context for testing task performance: stacked placement

Scene specification
```
(AN OBJECT (TYPE KITCHEN-MODEL))
(AN OBJECT (TYPE PLATE)
            (AT (A LOCATION (ON (AN OBJECT (NAME SINK-TABLE))))))
(AN OBJECT (TYPE PLATE)
            (AT (A LOCATION (ON (AN OBJECT (NAME SINK-TABLE))))))
```
Plan to run
```
(ENABLE-LOCATION-GENERATION STACKING-ON)
(PERFORM
 (AN ACTION (TYPE TRANSPORT)
            (ACTEE (AN OBJECT (TYPE PLATE) (QUANTITY :PLURAL)))
            (DESTINATION (A LOCATION (ON (AN OBJECT (TYPE TABLE)))))))
(PAR
 (PERFORM
  (AN ACTION (TYPE EATING)
             (ACTEE (AN OBJECT (TYPE PLATE)
                              (AT (A LOCATION (ON (AN OBJECT (TYPE TABLE)))))))
             (AGENT (AN OBJECT (TYPE HUMAN-MODEL) (NAME H-1)))))
 (PERFORM
  (AN ACTION (TYPE EATING)
             (ACTEE (AN OBJECT (RECOGNIZABLE-AS PLATE)
                              (AT (A LOCATION (ON (AN OBJECT (TYPE TABLE)))))))
             (AGENT (AN OBJECT (TYPE HUMAN-MODEL) (NAME H-2))))))
```
To check on simulation timeline
`(SUCCESS ?TL)`

An execution context will contain information about how to set up a simulation scenario, that is, what objects should exist and where they should be placed. The execution context also contains a list of tasks, expressed as action designators. The tasks may be performed by different agents and may vary in terms of what location designator resolution procedures they use. In particular, the execution context of table 1 uses the stacked-on designator solution candidate generator. Finally, the execution context contains a query to be performed on the simulation timeline, a record of the events in the simulation run. In this case, we want to test that all tasks in the execution context finished successfully.

For the present, we use simulated robots to represent humans because we do not yet have a human model available for the CRAM projection environment. This is because the primary concern in CRAM's development so far has been to reason about and produce robot actions. However, we think that giving the robot some representation of the actions humans expect to perform in the shared environment will help the robot understand its own tasks better and how it is supposed to *cooperate* with humans, so we will look into adding human models in the future.

4. Evaluation

We will now show results from simulation runs to interpret spatial relations. The execution context has 'table setting', done by one robot, as the main task, and we will allow either 'stacked' or 'individual' samplers for the 'on' relationship. The enabled task in the context will either be 'eating', to be done by two robots, or 'washing up', to be done by one robot, resulting in the following execution contexts: set the table for eating with plates stacked ("EatStk" in table 2), set the table for eating with plates individually placed ("EatInd"), or set the table for washing up ("WashStk" and "WashInd").

Table 2. Total simulation count, number of simulations ending in failure, and a breakdown of failure types for the various task contexts.

Task context	# Sims	# Failed Sims	Failures		
			Collision	Obj too far	Obj not Seen
EatStk	20	18	1	4	13
EatInd	20	6	0	6	0
WashStk	20	0	0	0	0
WashInd	20	0	0	0	0

We simulate each task context 20 times in projection. Object initial placement is randomized, and there is also a random element to the robot's generated placements and motions. Failures were logged for each scenario (cf. table 2).

As this record of failures shows, the 'stacked' vs 'individually placed' decision has no impact on the washing up task, which proceeds flawlessly. For eating however, if the plates are stacked, 18 of the simulations end in failure: either one of the robots reaching for the plate has its line of sight occluded by the other, or end effectors collide, or one of the robots cannot find a convenient place around the table from which to reach the plate. Individually placing plates on the table results in errors sometimes as well, but the error rate is lower (only 6 out of 20 simulations fail) and all these errors are alike: the "eating" robot chose to navigate to a place around the table that is too far from the plate.

The simulation therefore reveals some plausible geometric reasons for why it is a bad idea to leave plates stacked if several people expect to eat from them. This leads to filtering out some interpretations for the semantics of 'placing plates on', a filtering that depends on the task context and the geometric and behavioral constraints of the agents and the world they operate in. The way the robot interprets the 'near' relation can also

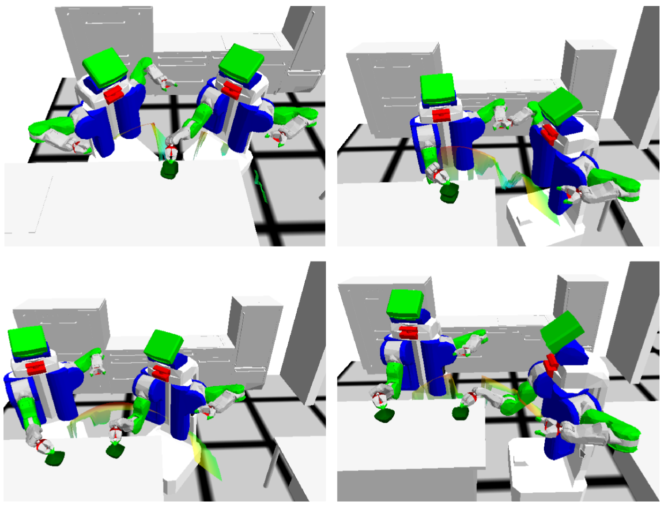

Figure 2. Robot configurations at projection end. Above: example end configurations for the eating from stacked plates scenarios; below: eating from individual plates.

be improved by the simulated experience, since it appears that the sampler for locations near the table is liable to produce locations that are too distant to actually reach items near the table. Such points in the costmap should have their probability density reduced, so that they and their neighbors are less likely to be chosen in the future.

In a similar fashion, other simulation scenarios can be created and evaluated. In the next example we look at a scenario to see how an action is performed affects some quality metric. Consider the instruction "take the cups to the table using the tray", which in GUM terms is an AffectingAction where the actor causes itself to move so as to transport some items to a destination GeneralizedLocation [7]. An instrument to assist the transport is specified, which results in adapting the transport program we generate to first place the transported items into (or, in this case, on) an auxiliary container first.

However, because the tray has a limited capacity, several trips between the source and destination tables will be needed. One could look at some metric of quality for the task completion. Optimality may be hard to reach, but one can keep track of the best simulated results as a good enough substitute. Projection does not measure time, so we use distance traveled by the robot base as a proxy.

In our particular case, the robot has to bring 4 cups to 4 locations on a destination table (see Figure 3). But, only 3 cups may be placed on the tray. It would seem a good strategy to always use the tray at full capacity, but in this case the robot will do an extra trip around the table, because the cup destination locations are spread out. It is often the case that what is a better approach to a task will depend subtly on geometric aspects of the environment the robot operates in, geometric aspects that linguistic instructions do not attempt to capture. Simulation then comes in as a way to explore possible behaviors, filter out inappropriate ones, and seek good-enough solutions for the task at hand.

5. Related work

Knowledge modeling for robotics presents a heterogeneous mix of representation and reasoning methods, as seen in the KnowRob system [32,31]. Capturing more formally

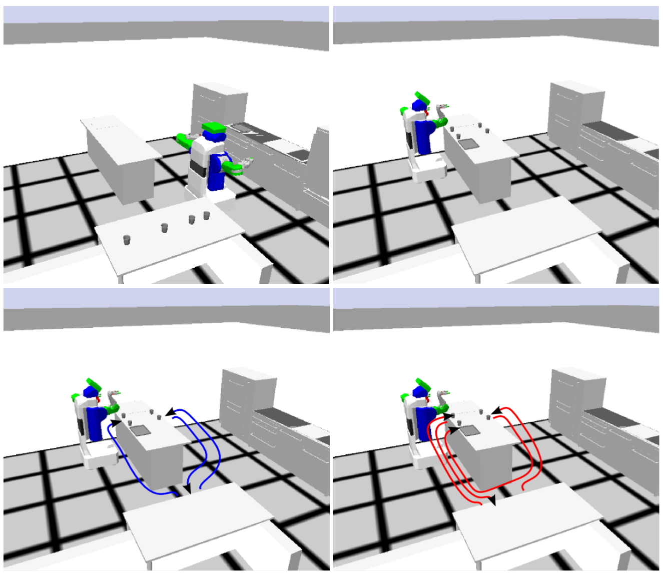

Figure 3. Transporting cups scenario: start and ending configurations (top row left and right respectively). Trips taken by the robot (bottom row): transporting 2 then 2 cups with tray (left); transporting 3 then 1 cup with tray (right).

the properties of hybrid reasoning systems for robotics has also been explored [6].

There has also been much research on interpreting language to programs. Some such translations are fairly straightforward, when the natural language itself resembles pseudocode, including some control structures [18]. The TellMeDave system [19] learns associations between instructions and human user activity logs. Markov Logic Networks are used in the PRAC system [23] to formalize understanding as probabilistic inference: find the most likely program given the instruction as evidence. PRAC can also do coreference resolution and fill in some missing information, such as a tool to use for an action.

None of the language understanding systems above take context into account, and operate purely on the semantics of the linguistic instruction. Interpretation filtering based on checking the feasibility of a plan in a STRIPS domain has been researched [26], but more general contextualization approaches have not been presented yet.

Some sensitivity to the environment has been shown in a logic-based system [10], where an instruction, translated into a first order logic statement, is verified for consistency with a first order logic description of the environment. Logics based approaches however may not scale well to everyday activities, since formalizations for these require extremely expressive and complex logics [12]. Instead, several levels of abstraction should be employed in a heterogeneous manner [5], to allow off-loading complexity to modules better suited to particular tasks, such as we do here with simulation. Support for heterogeneous knowledge modeling has already received attention [22,24].

Linguistic semantics is also embracing simulation as a necessary component of understanding, as seen in “Embodied Construction Grammars”[9], but the simulations considered there are rarely detailed enough for the execution of actual tasks on a robot.

6. Conclusions

We have described a simulation-based approach to contextualization for robotics and demonstrated how it can be used to interpret spatial relations appropriately to a task context, and improve robot behavior.

In the future we will be looking at using the ontological modeling to provide more flexible translations from semspecs to programs. Notions such as image schemas [15, 30], especially if ontologically formalized [14], may help here. We are also looking at retaining simulation results as a body of experiential knowledge so that rules for behavior can be learned, and help robot reactions to become as reflexive as human decisions are.

Acknowledgments

The research reported in this paper was supported by the German Research Foundation (DFG), as part of the Collaborative Research Center (Sonderforschungsbereich) 1320 "EASE - Everyday Activity Science and Engineering", University of Bremen (`http://www.ease-crc.org/`). The research was conducted primarily within subproject P01: 'Embodied Semantics for the Language of Action and Change'.

References

[1] Arbib, M.A., Gasser, B., Barrès, V.: Language is handy but is it embodied? Neuropsychologia **55**, 57–70 (2014)

[2] Barsalou, L.W.: Situated simulation in the human conceptual system. Language and Cognitive Processes **18**(5/6), 543–562 (2003)

[3] Bateman, J.A.: Language and Space: a two-level semantic approach based on principles of ontological engineering. International Journal of Speech Technology **13**(1), 29–48 (2010). https://doi.org/10.1007/s10772-010-9069-x

[4] Bateman, J.A.: Ontologies of Language and Language Processing. In: Poli, R., Healy, M., Kameas, A. (eds.) Theory and Applications of Ontology: Computer Applications, pp. 393–410. Springer, Dordrecht, Heidelberg, London and New York (2010)

[5] Bateman, J.A.: Space, Language and Ontology: A Response to Davis. Spatial Cognition & Computation **13**(4), 295–314 (2013). https://doi.org/10.1080/13875868.2013.808491

[6] Bateman, J.A., Beetz, M., Beßler, D., Bozcuoglu, A.K., Pomarlan, M.: Heterogeneous ontologies and hybrid reasoning for service robotics: The ease framework. In: Third Iberian Robotics Conference. ROBOT '17, Sevilla, Spain (2017)

[7] Bateman, J.A., Hois, J., Ross, R.J., Tenbrink, T.: A linguistic ontology of space for natural language processing. Artificial Intelligence **174**(14), 1027–1071 (September 2010), `http://dx.doi.org/10.1016/j.artint.2010.05.008`

[8] Beetz, M., Jain, D., Mösenlechner, L., Tenorth, M., Kunze, L., Blodow, N., Pangercic, D.: Cognition-enabled autonomous robot control for the realization of home chore task intelligence. Proceedings of the IEEE **100**(8), 2454–2471 (2012)

[9] Bergen, B.K., Chang, N.: Embodied Construction Grammar in simulation-based language understanding. In: Östman, J.O., Fried, M. (eds.) Construction Grammar(s): Cognitive and Cross-Language Dimensions, pp. 147–190. Johns Benjamins, Amsterdam (2005)

[10] Bos, J., Oka, T.: A spoken language interface with a mobile robot. Artificial Life and Robotics **11**(1), 42–47 (2007)

[11] Coventry, K.R., Garrod, S.C.: Saying, seeing and acting. The psychological semantics of spatial prepositions. Essays in Cognitive Psychology series, Psychology Press, Hove, UK (2004)

[12] Davis, E.: Qualitative spatial reasoning in interpreting text and narrative. Spatial Cognition & Computation **13**(4), 264–294 (2013). https://doi.org/10.1080/13875868.2013.824976, `https://doi.org/10.1080/13875868.2013.824976`

[13] Gennari, S.P.: Representing motion in language comphrension: Lessons from neuroimaging. Language and Linguistics Compass **6**(2), 67–84 (2012)

[14] Hedblom, M.M., Kutz, O., Neuhaus, F.: Choosing the right path: Image schema theory as a foundation for concept invention. Journal of Artificial General Intelligence **6**(1), 21–54 (2015)

[15] Johnson, M.: The body in the mind. University of Chicago Press, Chicago, Il (1987)

[16] Kasper, R.T.: A flexible interface for linking applications to PENMAN's sentence generator. In: Hirschman, L. (ed.) Proceedings of the DARPA Workshop on Speech and Natural Language. Morgan Kaufmann, San Mateo, CA (1989), `http://www.cs.mu.oz.au/acl/H/H89/H89-1022.pdf`, available from ACL Anthology H89-1022

[17] Kaup, B., Lüdtke, J., Maienborn, C.: 'the drawer is still closed': Simulating past and future actions when processing sentences that describe a state. Brain & Language **112**, 159–166 (2010)

[18] Matuszek, C., Herbst, E., Zettlemoyer, L., Fox, D.: Learning to Parse Natural Language Commands to a Robot Control System, pp. 403–415. Springer International Publishing, Heidelberg (2013)

[19] Misra, D.K., Sung, J., Lee, K., Saxena, A.: Tell me dave: Context-sensitive grounding of natural language to manipulation instructions. In: Proceedings of Robotics: Science and Systems. Berkeley, USA (July 2014)

[20] Morris, C.W.: Foundations of the Theory of Signs. University of Chicago Press, Chicago (1938)

[21] Mösenlechner, L., Beetz, M.: Fast temporal projection using accurate physics-based geometric reasoning. In: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA). pp. 1821–1827. Karlsruhe, Germany (May 6–10 2013)

[22] Mossakowski, T., Codescu, M., Neuhaus, F., Kutz, O.: The Distributed Ontology, Modeling and Specification Language – DOL. In: Koslow, A., Buchsbaum, A. (eds.) The Road to Universal Logic, vol. I, pp. 489–520. Birkhäuser (2015), `http://www.springer.com/gp/book/9783319101927`

[23] Nyga, D., Picklum, M., Koralewski, S., Beetz, M.: Instruction Completion through Instance-based Learning and Semantic Analogical Reasoning. In: International Conference on Robotics and Automation (ICRA) (2017)

[24] OMG: The Distributed Ontology, Modeling, and Specification Language (DOL). Tech. rep., OMG (2015), `https://github.com/tillmo/DOL/raw/master/Standard/dol.pdf`

[25] Pomarlan, M., Bateman, J.A.: Robot program construction via grounded natural language semantics & simulation. In: Proceedings of the 17th Conference on Autonomous Agents and MultiAgent Systems. AAMAS '18 (2018)

[26] Pomarlan, M., Koralewski, S., Beetz, M.: From natural language instructions to structured robot plans. In: Kern-Isberner, G., Fürnkranz, J., Thimm, M. (eds.) KI 2017: Advances in Artificial Intelligence. pp. 344–351. Springer International Publishing, Cham (2017)

[27] Regier, T., Carlson, L.A.: Grounding spatial language in perception: An empirical and computational investigation. Journal of Experimental Psychology: General **130**(2), 273–298 (2001)

[28] Richardson, D.C., Spivey, M.J., Barsalou, L.W., McRae, K.: Spatial representations activated during real-time comprehension of verbs. Cognitive Science **27**(5), 767–780 (September 2003)

[29] Roy, D.: Semiotic schemas: A framework for grounding language in action and perception. Artificial Intelligence **167**, 170–205 (2005)

[30] Talmy, L.: The fundamental system of spatial schemas in language. In: Hampe, B. (ed.) From perception to meaning: image schemas in cognitive linguistics, pp. 37–47. Mouton de Gruyter, Berlin (2006)

[31] Tenorth, M., Beetz, M.: KnowRob – A Knowledge Processing Infrastructure for Cognition-enabled Robots. Int. Journal of Robotics Research **32**(5), 566 – 590 (April 2013)

[32] Tenorth, M., Jain, D., Beetz, M.: Knowledge Representation for Cognitive Robots. Künstliche Intelligenz **24**(3), 233–240 (2010)

[33] Winkler, J., Tenorth, M., Bozcuoglu, A.K., Beetz, M.: CRAMm – memories for robots performing everyday manipulation activities. Advances in Cognitive Systems **3**, 47–66 (2014)

[34] Zimmer, H.D., Speiser, H.R., Baus, J., Blocher, A., Stopp, E.: The use of locative expressions in dependence of the spatial relation between target and reference object in two-dimensional layouts. In: Freksa, C., Habel, C., Wender, K.F. (eds.) Spatial Cognition I - An interdisciplinary approach to representing and processing spatial knowledge, pp. 223–240. Springer (1998)