Representing And Reasoning with Functional Knowledge for Spatial Language Understanding

Kalyan Moy Gupta[1], Abraham R. Schneider[1], Matthew Klenk[2], Kellen Gillespie[1], and Justin Karneeb[1]

[1]Knexus Research Corporation; Springfield, VA 22153
[2] Naval Research Laboratory, Washington, DC 20745
firstname.lastname@knexusresearch.com |

Abstract. One of the central problems in spatial language understanding is the polysemy and the vagueness of spatial terms, which cannot be resolved by lexical knowledge alone. We address this issue by developing a representation framework for functional interactions between objects and agents. We use this framework with a constraint solver to resolve and recover the meanings of spatial descriptions for object placement tasks. We describe our approach in a virtual environment with an example of object placement task.

1 Introduction

Virtual scene (re)construction or object placement is a vital task in many practical applications such as background layout in 3-D animated movies, accident or crime scene simulation, and navigation maps for video game development. Using natural language (NL) commands can be a natural and efficient alternative to an otherwise effort intensive manual placement of objects in a virtual world (e.g., Coyne and Sproat, 2001; and Dupuy, 2001). However, machine understanding of natural language commands is notoriously difficult due to polysemy and vagueness of spatial terms. Only considering lexical semantic knowledge of spatial terms is clearly insufficient for this task; world knowledge and pragmatics must be considered for understanding language in a form that can be acted upon by autonomous agents.

Over the past couple of decades much research in spatial term semantics has focused on developing computational models that map utterances to semantics (e.g., Regier and Carlson, 2001; Coventry et al., 1994). Although, such research recognizes the need for pragmatic and functional knowledge about objects, the development of computational models for representing and using such knowledge has received little attention. To address this gap, we present a framework for representing world knowledge that can be effectively translated into spatial constraints to resolve vague and underspecified natural language commands. We present an algorithm that utilizes such knowledge for interpreting natural language commands and to perform valid least cost object placements.

We organize the remainder of this paper as follows. We explain the nature of linguistic underspecification in object placement tasks in the next section. We follow this with a description of our representation framework and an algorithm that performs linguistically commanded single object placement task. Next, we illustrate our approach with an example. Finally, we discuss the strengths and limitations of our approach and conclude the paper.

2 Vagueness in NL driven Object Placement

Consider the task of generating a static scene described by text utterances in a 3D virtual environment. For example, generating a scene with “a chair *in front of* the table” and subsequently placing a “printer *on* the table.” The desired rendering of the scene is shown in Figure 1.

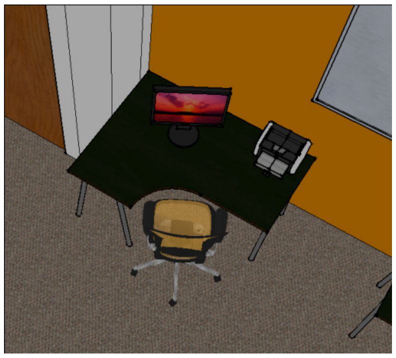

Figure 1. Example scene imagination based on linguistic description

The central issue in such a task is interpreting *vague* spatial prepositions such as `on` and `in-front-of` into *valid* object placements. The utterance “printer on the table” can only be judged as vague when attempting to place the printer in the World. For instance, the possible placements on the table are to the left, right, front, and behind the monitor. However, the placements in front and back of the monitor are functionally invalid for a human user. The utterance also does not specify the suitable orientation of the printer. Without such a specification, the printer could be oriented in numerous ways in relation to the monitor and the chair, only some of which would be valid. For example, the orientation shown in Figure 1 is a valid one. However, the orientations of the printer such as upside down or facing the wall would be invalid.

Clearly, functional knowledge of interaction between objects must be considered for generating valid placements. The question is what should be the content of such functional and world knowledge and how can it be utilized to recover the unspecified elements and generate a complete and valid specification for object placement. We answer this question in the next section.

3 Representation and Reasoning for Linguistically commanded Object Placement

Problem Task: Given a world, W, containing a set of objects, O, located in various places in the world and an underspecified linguistic command requesting to place a target object, o^t in W, find a location with the least *interaction cost* to place o^t. We return to the notion of interaction cost later in this section.

Functional Knowledge Representation. We introduce an autonomous agent, α, as the central element of a functional representation of objects, O, and their parts in the World. Given our goal of building agents that interact with humans, our representation encodes spatial constraints accordingly. We assume that α is human-like and interacts with objects using a set of *primitive actions* or perceived affordances (Gibson, 1977, Norman, 2002). We introduce a set of the following primitive actions:

1. *Reach*: the agent reaches for objects to manipulate and interact with them. Given our assumption that α is human-like, we subcategorize the *reach* interaction as follows:
   1.1. *Reach.Arm*: the agent reaches for objects with arms fully extended.
   1.2. *Reach.Forearm*: the agent reaches for objects with only the forearm extended.
   1.3. *Reach.Foot*: the agent reaches for object with its foot.
   1.4. *Reach.Assisted*: the agent reaches for objects with tools.
2. *See*: the agent obtains visual information from objects to perform reach actions. For an agent to see objects it must be oriented toward the objects. In certain situations the agent must be able to read the information present on the object. We represent this with the read action, a tighter constraint than see:
   2.1. *Read*: an agent reads the information present on the object such as signs or writing. Clearly, this can be subcategorized to read fine print, read normal print, read large print, read poster print etc.

We further assume that the agent performs these activities while it is located at certain places in W called *activity stations*, S. In addition, we assume that an agent has the following human-like *poses*; sitting, standing, and lying down.

We categorize the functional relation between objects into the following three types:

1. *Support*: this is a functional relation typically implied by the preposition "on" in English. For example, a table `supports` a printer and a printer is `supportedBy` a table.
2. *Contain*: this is a functional relation typically implied by the preposition "in". For example, a box `contains` the printer and a printer is `containedBy` a box.
3. *Group*: relates multiple objects into a spatial group. For example, a computer keyboard and display monitor may be related to each other by a spatial group relation.

Table 1. Example representation of functional interaction constraints for a Printer

Object/parts	Object	Agent		
	Interaction	Interaction	Pose	Activity Station
Printer/Parent		• Reachable.Arm • Visible	• Stand • Sit	Perimeter
Control panel	supportedBy(parent)	• Readable		
Connection panel	supportedBy(parent)	• Reachable.Arm		

Paper tray	containedBy(parent)
	contains(paper)
		• Reachable.Arm		

An object and its various parts may solicit different functional interaction constraints for agents. A representation of functional interaction constraints for a printer is shown in the Table 1. We assume a canonical geo-orientation for the printer, that is, it is upright. The table specifies that the printer control panel must be readable to the agent, for example.

We introduce the notion of a *possible interaction space, PIS*, for an agent at an activity station (e.g., see Kurup & Cassimatis, 2010). As a simplification in this paper, we assume that the possible interaction space is a two dimensional region. Figure 2 shows the PIS with reachability, visibility, readability spaces.

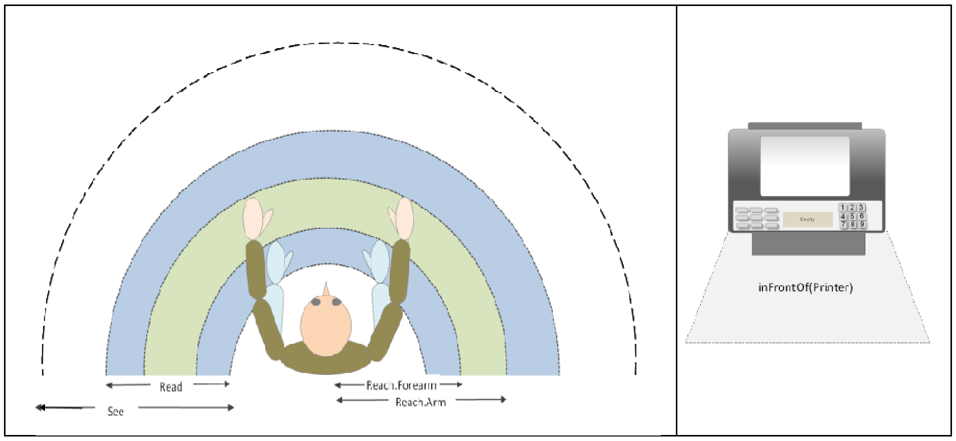

Figure 2. Possible interaction spaces for agent α and possible linguistically constrained space for infrontOf(Printer)

We introduce the notion of a *possible linguistically constrained space,* $PLCS$*,* as well. For example, Figure 2 shows the region selected by the function `inFrontOf(Printer)`. The possible space resulting from multiple constraints is the intersection of individual possible spaces. We will use this approach in the linguistically commanded single object placement algorithm presented next.

Linguistically commanded single object placement algorithm.
The algorithm begins by generating the set of potential activity stations to identify the smallest subset of that satisfies the spatial constraints in the World. The functional interaction knowledge of the objects in the World is transformed into spatial constraints. Next, it uses possibility spaces to identify the candidate placements and selects the one with least cost. We detail these steps below:

Inputs
1. O, set of objects in W.
2. o^t, the target object to be placed (e.g., Printer).
3. *lcs,* linguistically expressed placement constraint (e.g., on the table).
4. KB, the functional interaction knowledge base containing the agent and object interaction knowledge covering all objects in W (O and o^t).
5. α^{psp} the possibility space parameter for α for which the minimal cost placement is to be performed.

Output
1. P, a set of placements with minimum cost of functional interaction for agent α.

Processing steps
1. *Find the smallest set of activity stations,* S^{min}, that satisfes the functional interaction constraints for all objects $o^i \in O$; the constraints are retrieved from the KB for a given category of object. The candidate activity stations for an object o^i are located around its perimeter.
2. *Set the candidate placements,* $CP = \phi$*, placement cost, pc*=0
3. *Set candidate stations,* $S^c = S^{min}$
4. *For each activity station* $s^j \in S^c$

a. *Compute Possible Placement Space,* PPS^s for the target object o^t as the intersection of PIS, possible interactions space at the activity station and the LCS, linguistically constrained space:

$$PPS^s = PIS \cap PLCS$$

b. *Generate candidate placements (cp) and compute their cost, c*: The candidate placements are possible placements PPS^s if it is not empty. Only those placements that satisfy all the interaction constraints of o^t without violating any of the existing constraints satisfied by s^j are retained. The candidate placements are a combination of locations and orientations. For simplicity, we only consider 4 orientations of o^t relative to the orientation of the agent at the activity station. The cost of a placement is 0 when one of the existing stations is used for the placement.

```
5.  Select the minimum cost placements $P$.
    IF $CP \neq \phi$ THEN
        Return minimum cost placements $P \subset CP$
    ELSE,
        Generate new activity stations ($S^{new}$) in the neighborhood of stations in $S^c$.
        set $S^c = S^{new}$, and
        IF $pc$ is 0 set $pc$=1, i.e., cost of placement increases with the number of activity stations.
        go to Step 4
End.
```

Example

Consider a world W that includes a `table` placed against a `wall` with a `monitor` on it. In addition, it includes a `chair` located in front of the monitor (see Figure 3). The placement agent receives a linguistic command to place a `printer`, o^t, in this world; "Place the printer on the table".

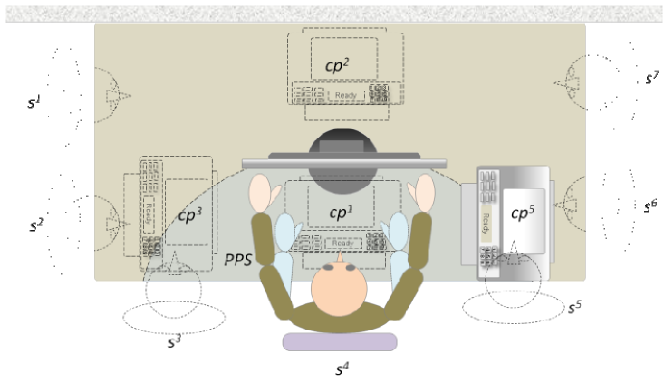

Figure 3. Place the printer on the table

We assume that this linguistic command (i.e., *lsc*) is interpreted into a semantic form and its $PLCS$ is computed, which is the entire surface of the table. We begin with step 1 to find S^{min}. The algorithm generates the potential activity stations in the world, for example, stations s^1 through s^7. It is easy to see that stations s^1, s^2, s^6 and s^7 do not satisfy the reachability and readability constraints for the monitor. Similarly, stations s^3 and s^5 fail to satisfy the readability constraint of the monitor. Notice that the alternative orientations of these stations would also fail on reachability constraints of various objects. Activity station s^4 is the only one that satisfies the reachability and readability constraints for the monitor and the reachability constraint of the chair (i.e., $S^{min}=\{ s^4\}=S^c$). We perform Step 4 and obtain PPS for s^4 shown in grey obtained by the intersection of PIS and $PLCS$ (on table). Since PPS is not empty, we create candidate placements cp^1 through cp^4. Although cp^1 satisfies the printer's reachability and readability constraints, it violates the monitor's readability constraints for station s^4. Similarly, cp^2 fails to satisfy the readability constraint for the printer. Note that reorienting cp^3 to face the agent will create a valid placement. The candidate placement cp^3 satisfies all the constraints and is a valid. Placement cp^4 is not in the PPS space and is shown here for illustration only. Our example illustrates how the algorithm using functional knowledge about object and agent interactions produces two valid placements for a printer given a highly underspecified placement directive.

4 Discussion

Recent research on spatial language understanding has pointed out the need for functional representations for understanding spatial utterances. For example, Coventry and Garrord (2004) present a functional geometric framework which includes geometric and dynamic kinematic routines, and object knowledge. Our approach also considers the dynamic interactions and object knowledge. However, we explicitly consider the role of an agent along with a very small set of interaction primitive affordances specialized for the object placement task. Further, we present an inferencing algorithm that utilizes the world knowledge to perform valid object placement. Lockwood (2009) also emphasizes the need for functional knowledge but focuses on structure mapping as a means learning functional knowledge for a scene labeling task. However, she did not include an interpretation method to recover meanings of underspecified utterances. In contrast, we manually encode the affordances to recover underspecified spatial semantics in object placement tasks. We intend to develop methods of acquiring the interaction knowledge in our future work.

Although, we demonstrated the use of functional knowledge for generating valid object placement, we did not consider the pragmatic and contextual elements such as plans, goals, and the situation of the agent requesting object placements. For instance, the directive "put the printer on the table" would carry different functional constraints with it if the requester were a mover in an office building or a warehouse instead of a worker in an office building. We plan to extend our models to include constraint selection based on the requesting agent's goals and intentions.

5 Conclusion

Interpretation of spatial descriptions and commands, such as those for an object placement, poses significant challenges due to polysemy and underspecification of spatial term semantics. To address this issue, we developed a functional interaction knowledge representation framework with a very small number of agent action primitives and object to object interaction primitives. We described a cost based constraint satisfaction algorithm for utilizing world knowledge for object placement. In our future work, we will implement and evaluate the performance our algorithm with varying number of objects in the scene and consider aspects of visual attention to resolve residual ambiguities and diectics (e.g., see Kelleher, 2003). Additionally, we will extend our approach to include the role of goals and intentions of the requesting agent in selecting the appropriate spatial constraints for object placement.

Acknowledgements
This research was supported by the Office of Naval Research. Matthew Klenk was supported by NRC Postdoctoral Research Fellowship. We thank the two anonymous reviewers for their comments and suggestions.

References

Coventry, K.R. & Garrod, S.C. (2004) *Saying, Seeing and Acting: The Psychological Semantics of Spatial Prepositions.* Hove: Psychology Press.

Coventry, K.R., Carmichael, R. & Garrod, S.C. (1994). Spatial prepositions, object-specific function and task requirements, *Journal of Semantics*, 11, 289-309.

Coyne, B., & Sproat, R., (2001) WordsEye: An automatic text-to-scene conversion system, SIGGRAP'01, *Proceedings of the 28th annual conference on computer graphics and interactive techniques*, pp. 487-496, New York, NY: ACM.

Dupuy, S. (2001). Generating a 3-D simulation of a car accident from a written description in natural language: The CARSIM system. *Proceedings of the Workshop on Temporal and Spatial Information Processing*, pp.1-8.

Gibson, K.J. (1977). The theory of affordances. In R. Shaw and J Bransford (Eds.), Perceiving, acting, and knowing: Toward and ecological psychology (pp. 67-82). Hillsdale, NJ: Erlbaum.

Kelleher, J.D. (2003). A perceptually based computational framework for the interpretation of spatial language, *Ph.D Thesis*, Dublin, Ireland: School of Computing, Dublin City University.

Kurup, U., & Cassimatis, N.L., (2010). Quantitative spatial reasoning for general intelligence, *Proceedings of the Third Conference on Artificial General Intelligence Conference*, pp. 1-6, Lugano, Switzerland: AGF.

Lockwood, K. (2009). Using analogy to model spatial language use and multimodal knowledge capture, *PhD Thesis*, Department of Computer Science, Evanston, IL: Northwestern University.

Norman, D. (2002). *The design of everyday things*, New York, NY: Basic Books.

Regier, T., & Carlson, L. (2001). Grounding spatial language in perception: An empirical and computational investigation. *Journal of Experimental Psychology: General, 130*, 273-298.