The Role of Event Simulation in Spatial Cognition

James Pustejovsky and Nikhil Krishnaswamy

Department of Computer Science, Brandeis University
415 South Street, Waltham, MA 02453, USA

Abstract. In this paper, we discuss the role that semantic simulations can play when reasoning about the spatial properties and consequences of events. To this end, we employ the modeling language VoxML to illustrate how 3D simulations of natural language expressions can inform different aspects of spatial reasoning associated with events and their participants. Specifically, we argue that multimodal simulations of events can reveal conceptual distinctions and presuppositions that are typically missed by traditional semantic modeling. This is due largely to the fact that linguistic utterances must be grounded within a spatial and temporal context, in order to satisfy the dynamic constraints inherent in the model. Furthermore, modeling with VoxML provides a language for formalizing the notion of affordance, by capturing an object's relationship between a situated agent and the grounded actions available in the environment and the object's habitat.

Keywords: Simulations, Spatial Reasoning, Affordances, Lexical Semantics.

1 Event Localization

While there has been considerable semantic, cognitive, and computational work devoted to the temporal interpretation of events as expressed in natural language, the spatial characteristics of events has had considerably less attention in linguistics than in its neighboring disciplines. Hence, while research in both spatial cognition and QSR has focused on frames of reference, object-activity affordance relations, and qualitative relations, linguistics has yet to interpret the spatial nature of events within the embedding context of an embodied agent, acting on differently afforded objects.

Where an event takes place is obviously dependent on the participants involved, and where they are themselves spatially situated. *Event localization* refers to the process of identifying the spatial extent of an event, activity, or situation (its *minimum embedding space*). The localization of an event depends on three major factors: (i) the dynamic structure of the event; (ii) its semantic type; and (iii) the specific role that the participants play in the event. Hence, localization can be defined as the computation of the minimum embedding space for the participants as they dynamically interact through the unfolding event.

In order to better appreciate the complexity of event localization, we will review some assumptions about the structure of events themselves. Events involve changes involving one or more objects over time, what we call the *object model.* Events that are causative introduce an additional level of involvement, called the *action model*. As pointed out in [7], within the embedding space, there are two subregions that can be identified: (a) the *event locus* is the region defined by the movements of the participants in the object model; (b) the *spatial aspect* involves a relative location, that is linguistically singled out.

Different types of events are, of course, located in space differently. If Mary gives John a book, then the "transfer of possession" event can be seen as that region involving the three event participants, Mary, John, and the book. If Mary sees a plane in the sky, then the "experiencer" event requires the stimulus (the plane) and the one experiencing (Mary). More difficult cases include events making *linguistic* reference to objects that are *spatially* not as salient to the actual localization of the event: in *The hurricane sank the ship*, the region of the entire hurricane is not as salient as where the ship actually sank. Similarly, when Mary visits her mother in Texas, there is no specification of where within Texas the event occurred, but we understand the implied space occupied by the event to be fairly limited in scope, e.g., a house or community. Hence, on purely linguistic terms, it is not so obvious what aspects of the participants are relevant to the computation of event localization. This is where multimodal simulation of an event can help reveal presuppositions regarding the spatial constraints and dynamics of event semantics.

2 Modeling with VoxML

The modeling language VoxML (Visual Object Concept Markup Language) [8] forms the scaffold used to link lexemes to their visual instantiations, termed the "visual object concept" or *voxeme*. In parallel to a lexicon, a collection of voxemes is termed a *voxicon*. There is no requirement on a voxicon to have a one-to-one correspondence between its voxemes and the lexemes in the associated lexicon, which often results in a many-to-many correspondence. That is, the lexeme *plate* may be visualized as a [[SQUARE PLATE]], a [[ROUND PLATE]], or other voxemes, and those voxemes in turn may be linked to other lexemes such as *dish* or *saucer*.

Each voxeme is linked to either an object geometry, a program in a dynamic semantics, an attribute set, or a transformation algorithm. VoxML treats objects and events in terms of a dynamic event semantics, Dynamic Interval Temporal Logic (DITL) [9]. The advantage of adopting a dynamic interpretation of events is that we can map linguistic expressions directly into simulations through an operational semantics [3, 4]. VoxML is used to specify the information beyond that which is inferable from the geometry, DITL, or attribute properties. VoxSim [2], the semantically-informed simulation environment built on the VoxML platform, does not rely on manually-specified categories of objects with identifying language, and instead procedurally composes the properties of voxemes in parallel with the lexemes they are linked with.

An OBJECT voxeme's semantic structure provides *habitats*, which are situational contexts or environments conditioning the object's *affordances*, which may be either "Gibsonian" or "Telic" *affordances* [1, 5, 6]. A habitat specified how an object typically occupies a space. When we are challenged with computing the embedding space for an event, the individual habitats associated with each participant in the event will both define and delineate the space required for the event to transpire. Affordances are used as attached behaviors, which the object either facilitates by its geometry (Gibsonian) or purposes for which it is intended to be used (Telic). For example, a Gibsonian affordance for *cup* is "grasp," while a Telic affordance is "drink from." This allows procedural reasoning to be associated with habitats and affordances, executed in real time in the simulation (VoxSim), inferring the complete set of spatial relations between objects at each state and tracking changes in the shared context between human and computer. Thus, simulation becomes a way of tracing the consequences of linguistic spatial cues through the narrative structure of an event.

3 Spatial Reasoning with VoxML

3.1 Enforcing Pre- and Postconditions

A VoxML entity's interpretation at runtime depends on the other entities it is composed with (be they objects or other relations), and their properties. One such canonical example would be placing an object [[knife]] in an [[in]] relation with another object [[mug]].

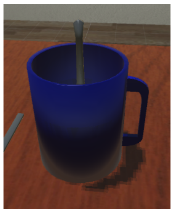

Fig. 1. [[knife in mug]]

The mug has an intrinsic top, which is oriented with the upward Y-axis of the world or embedding space. The VoxML denotation for this is $\{align(Y, \mathcal{E}_Y), top(+Y)\}$. The mug is also a concave object, and the mug's geometry (the [[cup]], excluding the handle) has reflectional symmetry across its inherent (object-relative) XY- and YZ-planes, and rotational symmetry around its inherent Y-axis such that when the object is in situated in its inherent *top* habitat, its Y-axis is parallel to the world's. From this we can infer that the *opening* (e.g., access to the concavity) must be along the Y-axis. Thus in order to compose an [[in]] relation with various types of objects, we have the following options, expressed as the RCC relation(s) (*E*xternally *C*onnected, *P*artial *O*verlap, *T*angential *P*roper *P*art) resulting from $rel(x, concavity\_type)$ (Fig. 2).

	Concave	**Non-concave**
[[in]]	EC	PO, TPP
[[on]]	EC	EC

Fig. 2. Relation × concavity composition ([[in]] vs. [[on]])

In order to produce the EC result required by the $put(x, in(y))$ event encoding, while maintaining contact with the object's concave geometry, the placed object x must *fit inside* the concave object y. In the case of the mug, it can be reasoned as shown that its concavity opens along the Y-axis, so any computation reasoner must also determine that the object to be placed within it can fit in that same orientation. In the case of a knife, normally lying flat on a surface, somewhere flush with the world's XZ-plane, simply placing it at the point where it would be EC with the mug would also cause it to interpenetrate the mug's sides inappropriately, and so the knife must first be turned (rotated) to align with the mug's opening. The requirements on the simulation of the *put knife in cup* event requirements enforce the resulting state of this "turn" action as a precondition which is otherwise not expressed in the language.

Relations created by events persist also after the completion of the event, and so too must they persist in event simulation.

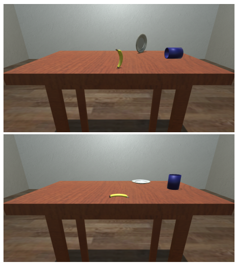

Fig. 3. Objects in unnatural (T) and natural (B) positions

Fig. 3 shows a number of objects at similar locations but in one case in orientations that, due to the effects of physics, would be considered "unstable" after the completion of a placement event. Object knowledge about thinks like *shape of cup*, *top of plate*, *default position of banana* mean that human observers can judge the top image to be unsatisfactory results of placement events and the bottom image to be more prototypical, due to the human ability to *simulate* what the result of an event in a given environment likely will be.

3.2 Mapping to Implementations

Computational event simulation requires a mapping from the formal structure of relation calculi to a computational implementation. This requires three components: 1) the mapping from a formal label to a computable instruction, 2) axiomatic closure over calculated relations for inference, and 3) adjustment for point of view. Let us examine the VoxML typing of the relation [[IN_FRONT]]:

$$\begin{bmatrix} \textbf{in\_front} \\ \text{TYPE} = \begin{bmatrix} \text{ARG} = \textbf{x:physobj} \\ \text{MAPPING} = \textbf{dimension(n):n} \\ \text{ORIENTATION} = \begin{bmatrix} \text{SPACE} = \textbf{pov} \\ \text{AXIS} = \textbf{+Z} \end{bmatrix} \end{bmatrix} \end{bmatrix}$$

[[IN_FRONT]] takes an n-dimensional region (typicall occupied by an object x and returns another entity of n dimensions (in 3D simulation, naturally n=3). A qualitative spatial representation of this might represent the comparison between these regions as $\texttt{max(x}_z\texttt{)} \geq \texttt{min(y}_z\texttt{)}$ (within ϵ), whereas an implementation of rectangle algebra in 3D space may need to further specify this as *greater_than*: $\texttt{max(x}_z\texttt{)} > \texttt{min(y}_z\texttt{)}$ or as *greater_than* $\wedge$ *meets* (i.e., [[IN_FRONT]] $\wedge$ [[TOUCHING]]): $\texttt{max(x}_z\texttt{)} > \texttt{min(y}_z\texttt{)} \wedge |\texttt{min(y}_z\texttt{)-max(x}_z\texttt{)}| < \epsilon$. Axiomatically, [[IN_FRONT]] exists in a pair with [[BEHIND]] : $in\_front(x, y) \rightarrow behind(y, x)$. Creating [[IN_FRONT]] must also automatically create the [[BEHIND]] inverse. This type of closure also reflects affordance distinctions of the type $put(x, in(y))$ results in $contain(y, x)$ so creating one type of relation ([[IN]]) also creates the other ([[CONTAIN]]), and vice versa. In *the hurricane sank the ship* from Section 1, the embedding space of the event contains the hurricane, which contains the ship, while the *locus* of the event is that region identified by the movement(s) within the object model of the event; i.e., from "floating on the surface of the water" to "being under water". This singles out the fact that the *sinking* event contains the most relevant information for localizing the entire event.

Finally, since linguistic expressions may be ambiguous based on the relative situatedness of the observers (i.e., the well-known problem of "my left" vs. "your left), event simulation in 3D requires reasoning over various types of spaces: *world* (absolute), *object*-relative (as discussed in Section 3.1), and point-of-view relative, as in this example, where the "in front" quality of the relation is relative to an observer, where if an object in front observer A may be to the left of observer B, based on the observers' relative position and frames of reference.

References

1. Gibson, J.J., Reed, E.S., Jones, R.: Reasons for realism: Selected essays of James J. Gibson. Lawrence Erlbaum Associates (1982)
2. Krishnaswamy, N., Pustejovsky, J.: VoxSim: A visual platform for modeling motion language. In: Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers. ACL (2016)
3. Miller, G., Charles, W.: Contextual correlates of semantic similarity. Language and Cognitive Processes **6(1)**, 1–28 (1991)
4. Miller, G.A., Johnson-Laird, P.N.: Language and perception. Belknap Press (1976)
5. Pustejovsky, J.: The Generative Lexicon. MIT Press, Cambridge, MA (1995)
6. Pustejovsky, J.: Dynamic event structure and habitat theory. In: Proceedings of the 6th International Conference on Generative Approaches to the Lexicon (GL2013). pp. 1–10. ACL (2013)
7. Pustejovsky, J.: Where things happen: On the semantics of event localization. In: Proceedings of the IWCS 2013 Workshop on Computational Models of Spatial Language Interpretation and Generation (CoSLI-3). pp. 29–39 (2013)
8. Pustejovsky, J., Krishnaswamy, N.: Voxml: A visualization modeling language. Proceedings of LREC (2016)
9. Pustejovsky, J., Moszkowicz, J.: The qualitative spatial dynamics of motion. The Journal of Spatial Cognition and Computation (2011)