Inferring Intention Through State Representations in Cooperative Human-Robot Environments

Craig Schlenoff[a, b], **Anthony Pietromartire**[a, b],
Dr. Zeid Kootbally[a], **Dr. Stephen Balakirsky**[a],
(craig.schlenoff, stephen.balakirsky, zeid.kootbally,
pietromartire.anthony) @nist.gov

Dr. Sebti Foufou[b, c]
sfoufou@qu.edu.qa

[a]National Institute of Standards
and Technology (NIST)
100 Bureau Drive, Stop 8230
Gaithersburg MD 20899 USA

[b]University of Burgundy
LE2i Lab,
Dijon, France

[c]Computer Science and
Engineering Department
Qatar University, Doha Qatar

ABSTRACT

In this chapter, we describe a novel approach for inferring intention during cooperative human-robot activities through the representation and ordering of state information. State relationships are represented by a combination of spatial relationships in a Cartesian frame along with cardinal direction information. The combination of all relevant state relationships at a given point in time constitutes a state. A template matching approach is used to match state relations to known intentions. This approach is applied to a manufacturing kitting operation[1], where humans and robots are working together to develop kits. Based upon the sequences of a set of predefined high-level state relationships that must be true for future actions to occur, a robot can use the detailed state information presented in this chapter to infer the probability of subsequent actions. This would enable the robot to better help the human with the operation or, at a minimum, better stay out of his or her way.

[1] Kitting is the process in which several different, but related items are placed into a container and supplied together as a single unit (kit)

1. INTRODUCTION

Humans and robots working safely and seamlessly together in a cooperative environment is one of the future goals of the robotics community (CCC, 2009). When humans and robots can work together in the same space, a whole class of tasks can be automated, ranging from collaborative assembly to parts and material handling to delivery. Keeping humans, who are within a robotic work cell safe, requires the ability of the robot to monitor the work area, infer human intention, and be aware of potential dangers to avoid them. Robots are under development throughout the world that will revolutionize manufacturing by allowing humans and robots to operate in close proximity while performing a variety of tasks (Szabo, 2011).

Proposed standards exist for robot-human safety (such as work performed by International Organization for Standardization (ISO) / Robotics Industries Association (RIA) 10218-2 Safety Requirements – Part 2: Industrial robot systems and integration), but these standards focus on robots adjusting their speed based on the separation distance between the human and the robot (Chabrol, 1987). In essence, as the robot gets closer to a detected human, the robot gradually decreases its speed to ensure that if a collision between the human and robot occurs, minimal damage will be caused. These standards focus on where the human is at a given point in time. It does not focus on where they are anticipated to be in the future.

A key enabler for human-robot safety in cooperative environments involves the field of intention recognition. Intention recognition involves the robot attempting to understand the intention of an agent (the human) by recognizing some or all of his/her actions (Sadri, 2011) to help predict the human's future actions. Knowing these future actions will allow a robot to plan in such a way as to either help the human perform his/her activities or, at a minimum, not put itself in a position to cause an unsafe situation.

In this chapter, we present an approach to representing state information in an ontology for the purpose of ontology-based intention recognition. An overview of the intention recognition approach can be found in (Schlenoff, 2012a). In this context, we adopt the (Tomasello, 2005) definition of intention as "a plan of action the organism chooses and commits itself to in pursuit of a goal – an intention thus includes both an action plan as well as a goal." We also distinguish states from state relationships. In this context, we define a state as a set of properties of one or more objects in an area of interest that consist of specific recognizable configurations and or characteristics. A state relationship is a specific relation between two objects (e.g., Object 1 is on top of Object 2). A set of all relevant state relationships in an environment composes a state. This approach to intention recognition is different than many ontology-based intention recognition approaches in the literature (as described in the next section) as they primarily focus on activity (as opposed to state) recognition and then use a form of abduction to provide explanations for observations. We infer detailed state relationships using observations based on Region Connection Calculus 8 (RCC8) (Randell, 1992) and then combine these observations using the Semantic Web Rule Language (SWRL) (W3C_Member_Submission, 2004) to infer the overall state relationships that are true at a given time. Once a sequence of state relationships has been determined, we will use probabilistic procedures to associate those states with likely overall intentions to determine the next possible action (and associated state) that is likely to occur. This chapter focuses on the way that states are represented in the ontology and how intentions can be inferred from them.

We start by providing an overview of intention recognition efforts in the literature as well as various approaches for ontology-based state representation. We then present two evaluations, one in

intention recognition and one in activity recognition, and show the difference in performance between the two. After that, we show a novel approach to state relation representation and ordering to describe intentions using RCC-8 and SWRL. We then describe the manufacturing kitting domain and provide a detailed scenario showing how the approaches can be used to represent state information in this domain. We then show how we can use the ordering of these states and state relationships to help to identify intentions that are occurring in the environment. We conclude the chapter by showing advantages of the state-based recognition approach as compared to other approaches in the literature.

2. INTENTION RECOGNITION AND STATE REPRESENTATION RELATED WORK

Intention recognition traditionally involves recognizing the intent of an agent by analyzing some of, or all of, the actions that the agent performs. Many of the recognition efforts in the literature are composed of at least three components: (1) identification and representation of a set of intentions that are relevant to the domain of interest, (2) representation of a set of actions that are expected to be performed in the domain of interest and the association of these actions with the intentions, (3) recognition of a sequence of observed actions executed by the agent and matching them to the actions in the representation. (Sadri, 2011)

There have been many techniques in the literature applied to intention recognition that follow the three steps listed above, including an ontology-based approach (Jeon, 2008), multiple probabilistic frameworks such as Hidden Markov Models (Kelley, 2008) and Dynamic Bayesian Networks (Schrempf, 2005), utility-based intention recognition (Mao, 2004), and graph-based intention recognition (Youn, 2007). In this chapter, we focus on ontology/logic-based approaches.

In many of these efforts, abduction has been used as the underlying reasoning mechanism in providing hypotheses about intentions. In abduction, the system "guesses" that a certain intention could be true based on the existence of a series of observed actions. For example, one could guess that someone may have watered the lawn if the lawn is wet. There may be other possible explanations for why the lawn is wet (e.g., it rained) but the fact that someone watered the lawn could be the most probable explanation given the circumstances. As more information is learned and activities are performed, probabilities of certain intentions can be refined to be consistent with the observations.

Intention recognition is a rich and challenging field. As shown above, often multiple competing hypotheses are possible regarding the intentions of an observed agent or to describe the situation in an environment. Choosing between these hypotheses is one challenge, but there are many others. One challenge is that circumstances, including the adversarial nature of the observed agent, may afford only partial observability of the actions. In other words, the scene may be occluded by other objects so that it can not be seen in its entirety, or the agent performing the action may purposefully hade their actions or intentions from the observer. Furthermore, would-be-intruders and would-be attackers may even deliberatively execute misleading actions to throw off the intention recognition system.

Another challenge is the case where the agent may have multiple intentions that they are performing at the same time and may interleave the execution of their actions, or the case where the actor is concurrently trying alternative plans for achieving the same intention. As another example, intention recognition becomes more difficult when attempting to interpret the actions of cognitively impaired individuals who may be executing actions in error and with confusion, for example in the case of Alzheimer patients (Roy, 2007). Additional complications arise when attempting to analyze the actions and intentions of multiple, cooperating agents (Sukthanker, 2001)

Much research in the intention recognition field focuses on pruning the space of hypotheses. In a given domain, there could be many possible intentions. Based on the observed actions, various techniques have been used to eliminate improbable intentions and assign appropriate probabilities to intentions that are consistent with the actions performed. Some of the ways that efforts have done this is by assigning weights to conditions of the rules used for intention recognition as a function of the likelihood that those conditions are true (Pereira, 2009). For example, it may be unlikely that a person is using an umbrella outside unless it is raining or very sunny, thus the cost associated with this condition would be very high since the probability of this happening would be low.

Once observations of actions have been made, different approaches exist to match those observations to an overall intention or goal. For example, in (Mulder, 2003), the authors use existentially quantified observations (not fully grounded observations) to match actions to plan libraries. Mulder can handle situations when they see an action occur (e.g., opening a door), without seeing or knowing who performed that action. Other approaches have focused on building plans with frequency information, to represent how often an activity occurs (Jarvis, 2005). The rationale behind these approaches is that there are some activities that occur very frequently and are often not relevant to the recognition process (e.g., a person cleaning his/her hands). These frequently-occurring activities can be mostly ignored, and only activities that are less commonly performed can be considered. In (Demolombe, 2006), the authors combine probabilities and situation calculus-like formalization of actions. In particular, Demolombe not only defines the actions and sequences of actions that constitute an intention, they also state which activities cannot occur for the intention to be valid. For example, if the intention was to drive a car, the activity may be to open the door, get into the car, turn on the engine, release the emergency brake, and take the car out of park. Demolombe may also include that an activity cannot be to turn the car off after it is turned on and before the car is taken out of park.

All of these approaches have focused on the activity being performed as the primary basis for observation and the building block for intention recognition. However, as noted in (Sadri, 2011), activity recognition is a very hard problem and one that is far from being solved. There has been limited success in using Radio Frequency Identification (RFID) readers and tags attached to objects of interest to track their movement with the goal of associating their movement with known activities. For example, in (Philipose, 2005), the authors describe the process of making tea as a three step process involving using a kettle, getting a box of tea bags, and adding some combination of milk, sugar, or lemon. Each of these activities is identified by a user wearing a special set of gloves that can read RFID tags on objects of interest. However, this additional hardware can be inhibiting and unnatural. Recognizing and representing states as opposed to actions can help to address some of the issues involved in activity recognition (e.g., the location of the tea bag box and the milk carton with respect to the tea cup) will be the focus of the rest of this paper.

State representation is documented in the literature, although it has not been used (to the authors knowledge) for ontology-based intention recognition. An important aspect of an object's state is its spatial relationships to other objects. In (Bateman, 2006), an overview is given that describes the way that spatial information is represented in various upper ontologies including the Descriptive Ontology for Linguistics and Cognitive Engineering (DOLCE), Cyc, the Standard Upper Merged Ontology (SUMO), and Basic Formal Ontology (BFO). The conclusion of this work is the identification of a list of high-level requirements that were necessary for any spatial ontology, including:

1. A selection of an appropriate granular partition of the world that picks out the entity that we wish to locate with respect to other entities.
2. A selection of an appropriate space region formalization that brings out or makes available relevant spatial relationships.

3. A selection of an appropriate partition over the space region (e.g., RCC8, qualitative distance, cardinal direction).
4. The identification of the location of the entity with respect to the selected space region description.

Bateman ended up using a variation of the DOLCE ontology, but there is no mention in the literature about the detailed spatial relations that were developed as part of this effort.

Region Connection Calculus 8 (RCC-8) (Wolter, 2000) is a well-known and cited approach for representing the relationship between two regions in Euclidean space or in a topological space. There are eight possible relations, including disconnected, externally connected, partially overlapping, etc. However, RCC8 only addresses these relationships in two-dimensional space. There have been other approaches that have tried to extend this into a region connected calculus in three-dimensional space while addressing occlusions (Albath, 2010). There have also been other approaches to develop calculi for spatial relations. FlipFlop calculus (Ligozat, 1993) describes the position of one point (the referent) in a plane with respect to two other points (the origin and the relatum). Single Cross Calculus (SCC) (Freksa, 1992) is a ternary calculus that describes the direction of a point (C - the referent) with respect to a second point (B - the relatum) as seen from a third point (A - the origin) in a plane. Double Cross Calculus (DCC) (Freksa, 1992) extends SCC by allowing one to also determine the relative location of point A with respect to point B (in addition to point B with respect to point A as in SCC). Coarse-grained Dipole Relation Algebra (Schlieder, 1995) describes the orientation relation between two dipoles (an oriented line segment as determined by a start and end point). Oriented Point Relation Algebra (OPRA) (Moratz, 2005) relates two oriented points (a point in a plane with an additional direction parameter) and describes their relative orientation towards each other. All of these approaches, apart from RCC8, focus on points and lines as opposed to regions. Also, despite the large variety of qualitative spatial calculi, the amount of applications employing qualitative spatial reasoning techniques is comparatively small (Wallgrun, 2006).

Throughout the rest of this paper, we will describe an approach for ontology-based state representation within the context of a typical manufacturing scenario.

3. STATE RECOGNITION VS. ACTIVITY RECOGNITION

One of the core assumptions of this work is that it is easier to perform state recognition in an environment as opposed to activity recognition. While there have been many efforts that have attempted to do each, they have primarily been self-evaluated in configurations and environments that have been most conducive to their approaches. There have, however, been some open, impartial competitions/evaluations in these areas (DARPA, 2012; Marvel, 2012). Below we describe a state recognition competition and an activity recognition evaluation performed by external, impartial parties. We will use the results of these competitions as benchmarks to characterize the state of the art in these two fields.

State recognition can encompass many things, including recognizing objects' color, size, shape, location, pose, as well of the identification of the object itself. Later in this chapter, we describe an approach that relies on the identification of an object as well as determining its location and pose to characterize its spatial relationships with other objects. As such, we will use the competition described

in Section 3.1 as the basis for the performance of state recognition technology since it is well aligned with the focus of this paper.

3.1. State of the Art in State Recognition

In 2011, as part of the International Conference on Robotics and Automation (ICRA) in Shanghai, China, a Solutions in Perception Challenge (SPC) was held (Newman, 2011). The purpose of establishing the SPC was to determine the current state of maturity for robotic perception algorithms. There are many algorithms that currently exist world-wide for identifying objects and determining their pose (location and orientation with respect to a coordinate frame), yet it is difficult to ascertain whether an algorithm could be applied to a given task or to know with confidence its robustness. In addition, efforts to develop these algorithms are being duplicated, but there is no convenient way of readily knowing what algorithms have already solved a particular aspect of the perception problem (Marvel et al., 2012). The SPC seeks to identify the best available perception algorithms that will be documented in the form of open source software to prevent duplication of development efforts and, in turn, accelerate the development of the next generation of perception algorithms.

The topic of the 2011 Challenge was single and multiple rigid object identification and 6 degree of freedom (6DOF) pose estimation in structured scenes. Competing teams were required to develop algorithms that could "learn" an arbitrary number of objects from the provided 3D point cloud data that had been augmented with the corresponding red-green-blue point color, and then to correctly identify and locate the same objects in a presented scene.

The data sets composing the training and evaluation sets were assembled using 16 machined aluminum artifacts representing commonly-encountered features of manufactured parts (Figure 1). Each artifact was created from a unique computer-aided design model, and was augmented with semi-Lambertian, optically-textured decals. The artifacts were first categorized into three classification groups based on perceived physical features. Group 1 consisted of objects with maximal heights greater than 2 inches. Group 2 objects were shorter than 2 inches, but had raised features that gave them non-level surfaces. And Group 3 objects were shorter than 2 inches, and had level surfaces. The artifacts were designed to be congruent with industrial assembly parts like automotive or aircraft components, and could be rigidly fixtured to a low-cost ground truth system (Figure 2).

Figure 1: Sample machined NIST artifacts used in the 2011 SPC, arranged randomly.

Figure 2: The ground truth fixture with an attached sensor and an artifact in the rotation plate

Evaluation of the submitted algorithms was broken into two distinct rounds. Round 1 consisted of image frames featuring only one artifact at a time, while the frames in Round 2 contained three artifacts each. Both rounds were composed of several sub-runs based on variations in object translation and rotation. Run 1 consisted of only object translations; Run 2 had only object rotations using the four fixture-based alignment holes; and Run 3 had a combination of translations and rotations. The teams were not aware of the composition of the data sets prior to the competition.

For each run in Round 1, a sample artifact was randomly selected from the three classification groups. Poses compliant with the Run-based transformation restrictions were then randomly selected for

each run, and the chosen artifacts were each applied to the same subset of transformations. For Round 2, one object from each classification group was randomly selected to form the test group. Random poses were generated for each run, with each artifact being assigned a different location on the base plate. There were a total of 399 frames for the contestants to assess.

Figure 3: Translation scores over all 399 frames

For each frame of data, the ground truth consisted of a set of one or more objects. A true positive count (*hits*, c_h) reflects an algorithm's ability to correctly identify when an object is in the scene. A non-zero false positive count (*noise*, c_n) indicates that an algorithm identified objects that were not actually present in the ground truth, and a non-zero false negative count (*misses*, c_m) implies that the algorithm could not correctly identify objects that were in the scene.

The true positive counts were tallied over all 399 frames. The competing teams correctly identified over 80 % of all objects over all frames (665 artifacts or more of the 831 present over all 399 frames).

Of the seven competing teams, two teams scored above 80% on the translation test (i.e., the estimated translation was within tolerance greater than 80 % of the time). In addition, three of the seven teams scored greater than 72 % on the rotation test (see **Figure 3**). The baseline score in the figures was the performance obtained by a Willow Garage test system used to develop the data sets and evaluation metrics.

3.2. State of the Art in Activity Recognition

Activity recognition can be roughly divided into two camps. The first focuses on sensor-based activity recognition where the person in the environment wears some type of sensor, which could be a smartphone, accelerometer, or other tracking device, that provides data as to where the person is and the motion s/he is taking. Examples of this include (Choudhury, 2008) and (Ravi, 2005). The second is vision-based activity recognition, where the behaviors of agents are characterized using videos taken by various cameras. Examples of this type of work can be found in (Bodor, 2003) and (Hoogs, 2008).

For the work described in this chapter, the focus is on approaches that do not require the agent to wear any external or internal device, as this is not reasonable or practical in an industrial setting. Therefore we will only consider vision-based activity recognition. In addition, as mentioned earlier in this section, we will limit our analysis to evaluations that were performed by an external evaluator (i.e., not the developers of the systems) to remove any conflict of interest and ensure an unbiased evaluation. In the next paragraph, we will describe one such evaluation and the subsequent results.

The Defense Advanced Research Projects Agency (DARPA) Mind's Eye Program[2] seeks to develop machine-based visual intelligence by automating the ability to learn generally applicable and generative representations of actions between objects in a scene directly from visual inputs, and then reason over those learned representations. The focus of this project is on the military domain, where Army scouts are commonly tasked with covertly entering uncontrolled areas, setting up a temporary

[2] http://www.darpa.mil/Our_Work/I2O/Programs/Minds_Eye.aspx (August 14, 2012)

observation post, and then performing persistent surveillance for 24 hours or longer. A truly "smart" camera would describe with words everything it sees and reason about what it cannot see. These devices could be instructed to report only on activities of interest, which would increase the relevancy of incoming data to users. Thus, smart cameras could permit a single scout to monitor multiple observation posts from a safe location.

To evaluate the activity recognition, the DARPA program identified 2,588 short vignettes (approximately 15 s to 20 s each) of 48 different activities. Each activity was represented by approximately 54 of the short vignettes. The activities that were explored are shown in Table 1.

Table 1: Verbs Used on the DARPA Mind's Eye Program

Approach	Close	Flee	Have	Open	Run
Arrive	Collide	Fly	Hit	Pass	Snatch
Attach	Dig	Follow	Hold	Pick up	Stop
Bounce	Drop	Get	Kick	Push	Take
Bury	Enter	Give	Jump	Put down	Throw
Carry	Exchange	Go	Leave	Raise	Touch
Catch	Exit	Hand	Lift	Receive	Turn
Chase	Fall	Haul	Move	Replace	Walk

The eight systems under evaluation had to determine if one or more of the verbs listed above were present in the vignettes. There were two metrics used. The first was precision which is defined as:

$$\boldsymbol{Precision} = \frac{VerbMatches}{VerbMatches + FalsePositives} \tag{1}$$

The precision results from each team are shown in Figure 4. Team scores ranged from 21 % to 60 %.

Figure 4: Precision Results from Mind's Eye Evaluation

For the second metric, the program used the Matthews Correlation Coefficient (MCC), which is a balanced measure of correlation which can be used even if the classes are of different sizes (in lieu of accuracy). The formula is:

$$\boldsymbol{MCC} = \frac{TP * TN - FP * FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}} \tag{2}$$

where TP are true positives, TN are true negatives, FP are false positives, and FN are false negatives.The results of applying this metric are very similar to the precision metrics, where systems scored between 12 % to 63 % when using this metric, with an average of approximately 35%.

In summary, in both cases, the system was given a scene or video and asked to characterize it by stating either the type and pose/orientation of an object or the existence of an activity. While it is impossible to perform a direct comparison between these two approaches, these examples provide a sampling of the type of performance that can be expected by each type of technology. Based on this analysis, it appears that state recogition systems are approximately 20 % (80 % vs. 60 %) more accurate than the activity recognition systems. This shows the benefit in using state recognition systems in performing intention recognition as opposed to using activity recognition systems.

4. REPRESENTING STATES AND SPATIAL RELATIONS

In this section, we describe an approach that uses RCC8 to model state relationships based on the relative position of objects in the environment. However, we will extend RCC8, which was initially developed for a two-dimensional space, to a three-dimensions space by applying it along all three planes (x-y, x-z, y-z). The frame of reference will be with respect to the fixed object (e.g., a worktable), with the z-dimension pointing straight upwards and the tabletop extending in the x- and y- dimension (with detailed orientation specific to the application). Each of the high-level state relationships (to be discussed later in the chapter) will have a set of logical rules that associate these RCC8 relations to them. These RCC8 relations, in theory, should easily allow a sensor system to characterize the corresponding state relations.

As mentioned earlier, RCC8 abstractly describes regions in Euclidean or topological space by their relations to each other. RCC8 consists of eight basic relations that are possible between any two regions:

- Disconnected (DC)
- Externally Connected (EC)
- Equal (EQ)
- Partially Overlapping (PO)
- Tangential Proper Part (TPP)
- Non-Tangential Proper Part (NTPP)
- Tangential Proper Part Inverse (TPPi)
- Non-Tangential Proper Part Inverse (NTPPi)

These are shown pictorially in Figure 5.

Figure 5: RCC8 Relations (Credit: http://en.wikipedia.org/wiki/RCC8)

RCC8 was created to model the relationships between two regions in two dimensions. In many domains, these relations need to be modeled in all three dimensions. As such, every pair of objects has a RCC8 relation in all three dimensions. To address this, we are prepending an x-, y- or z- before each of the RCC8 relations. For example, to represent the RCC8 relations in the x-dimension, the nomenclature would be:

- x-DC
- x-EC
- x-EQ
- x-PO
- x-TPP
- x-NTPP
- x-TPPi
- x-NTPPi

Similar nomenclature would be used in the y- and z- dimensions. The combination of all 24 RCC relations starts to describe the spatial relations between any two objects in the scene. However, more information is needed to represent the cardinal direction between any two objects. For example, to state that a worktable is empty (worktable-empty(wtable)), one needs to state that there is nothing on top of it. If we assume that the vertical dimension is the z-dimension, then saying that:

$$\text{z-EC}(\textit{wtable, obj1}) \tag{3}$$

(which intuitively means *obj1* is externally connected to the worktable in the z-dimension) is not sufficient because *obj1* could be either on top of or below the worktable. In other words, we need to represent directionality. We do this using the following Boolean operators:

$$\text{greater-x}(A,B) \tag{4}$$
$$\text{smaller-x}(A,B) \tag{5}$$
$$\text{greater-y}(A,B) \tag{6}$$
$$\text{smaller-y}(A,B) \tag{7}$$
$$\text{greater-z}(A,B) \tag{8}$$
$$\text{smaller-z}(A,B) \tag{9}$$

which intuitively means, in Equation 4, that the edge of the bounding plane (discussed later in Section 8) of object A is greater than (in the x-dimension in the defined frame of reference) the edge of the bounding plane of object B.

There are undoubtedly many other relationships that may be needed in the future to describe a scene of interest. These could include absolute locations and orientations of objects (x, y, z, roll, pitch, yaw) and relative distance (closer, farther, etc.). However, these spatial relations are sufficient for describing the manufacturing kitting example later in this chapter.

From these RCC8 spatial relations, we can define more complex spatial relations such as the ones below:

- **contained-in** – an object is enclosed in a second object from all sides
- **not-contained-in** - an object is not enclosed in a second object from all sides
- **partially-in** – an object is inside of another object in at least two dimensions but not fully contained in
- **in–contact-with** – touching at least one side and not contained with (i.e., toughing outer edges)
- **on-top-of** – the edge of the bounding plane of the one object is greater (in the z-dimension) than that of a second object
- **under** - the edge of the bounding plane of the one object is less (in the z-dimension) than a second object

And from these, we can define composite spatial relationships such as:

- **under and in contact with** – an object is both under and in contact with a second object
- **partially in and in contact with** – an object is inside of another object in at least two dimensions and touching the object in at least one dimension

Below, we formalize these spatial relationships by defining them using the RCC8 state representation. All of the formulas described below are represented by SWRL rules. Each atom in the SWRL rules references a predicate, which is represented by object properties in the ontology.

In natural language, Equation 10 below states that object 1 (obj1) is contained in object 2 (obj2) if obj1 is tangentially or non-tangentially a proper part of object2 in the x, y, and z-dimension. One can logically envision this by drawing two convex figures, and the first convex hull is completely inside of the second convex hull in all three dimensions, with it touching or not touching the second convex hull in all of the three dimensions.

$$\begin{gathered}\textbf{Contained-In}(\boldsymbol{obj1}, \boldsymbol{obj2}) \rightarrow \\ (\text{x-TPP}(obj1, obj2) \vee \text{x-NTPP}(obj1, obj2)) \wedge \\ (\text{y-TPP}(obj1, obj2) \vee \text{y-NTPP}(obj1, obj2)) \wedge \\ (\text{z-TPP}(obj1, obj2) \vee \text{z-NTPP}(obj1, obj2))\end{gathered} \tag{10}$$

$$\begin{gathered}\textbf{Not-Contained-In}(\boldsymbol{obj1}, \boldsymbol{obj2}) \rightarrow \\ \neg\text{Contained-In}(obj1, obj2)\end{gathered} \tag{11}$$

$$\begin{gathered}\textbf{Partially-In}(\boldsymbol{obj1}, \boldsymbol{obj2}) \rightarrow \\ \text{Not-Contained-In}(obj1, obj2) \wedge \\ ((\text{x-TPP}(obj1, \text{obj2}) \vee \text{x-NTPP}(obj1, obj2)) \wedge \\ (\text{y-TPP}(obj1, obj2) \vee \text{y-NTPP}(obj1, obj2))) \vee \\ ((\text{x-TPP}(obj1, obj2) \vee \text{x-NTPP}(obj1, obj2)) \wedge \\ (\text{z-TPP}(obj1, obj2) \vee \text{z-NTPP}(obj1, obj2))) \vee \\ ((\text{y-TPP}(obj1, obj2) \vee \text{y-NTPP}(obj1, obj2)) \wedge \\ (\text{z-TPP}(obj1, obj2) \vee \text{z-NTPP}(obj1, obj2)))\end{gathered} \tag{12}$$

In-Contact-With(*obj1*, *obj2*) → (13)
x-EC(*obj1*, *obj2*) ∨ y-EC(*obj1*, *obj2*) ∨ z-EC(*obj1*, *obj2*)

On-Top-Of(*obj1*, *obj2*) → (14)
greater-z(*obj1*, *obj2*) ∧ (x-EQ(*obj1*, *obj2*) ∨ x-NTPP(*obj1*, *obj2*) ∨ x-TPP(*obj1*, *obj2*) ∨ x-PO(*obj1*, *obj2*) ∨ x-NTPPi(*obj1*, *obj2*) ∨ x-TPPi(*obj1*, *obj2*)) ∧ (y-EQ(*obj1*, *obj2*) ∨ y-NTPP(*obj1*, *obj2*) ∨ y-TPP(*obj1*, *obj2*) ∨ y-PO(*obj1*, *obj2*) ∨ y-NTPPi(*obj1*, *obj2*) ∨ y-TPPi(*obj1*, *obj2*))

Under(*obj1*, *obj2*) → (15)
smaller-z(*obj1*, *obj2*) ∧ (x-EC(*obj1*, *obj2*) ∨ x-NTPP(*obj1*, *obj2*) ∨ x-TPP(*obj1*, *obj2*) ∨ x-PO(*obj1*, *obj2*) ∨ x-NTPPi(*obj1*, *obj2*) ∨ x-TPPi(*obj1*, *obj2*)) ∧ (y-EC(*obj1*, *obj2*) ∨ y-NTPP(*obj1*, *obj2*) ∨ y-TPP(*obj1*, *obj2*) ∨ y-PO(*obj1*, *obj2*) ∨ y-NTPPi(*obj1*, *obj2*) ∨ y-TPPi(*obj1*, *obj2*))

Under-And-In-Contact-With(*obj1*, *obj2*) → (16)
Under(*obj1, obj2*) ∧ In-Contact-With (*obj1*, *obj2*)

Partially-In-And-In-Contact-With(*obj1*, *obj2*) → (17)
Partially-In(*obj1*, *obj2*) ∧ In-With-Contact(*obj1*, *obj2*)

These spatial relationships will be used later in the chapter to define two manufacturing kitting intentions.

5. ORDERING STATES TO FORM INTENTIONS

In this work, an ordering of state relationships represents an intention. As such, we need a formal mechanism to allow for this ordering. To do this, we borrow some concepts that are described in OWL-S (Web Ontology Language – Services) (Martin, 2004). OWL-S is described on the website (http://www.w3.org/Submission/OWL-S/) as an ontology of services enabling a user and software agents to discover, invoke, compose, and monitor Web resources offering particular services and having particular properties. Though intended for web-based services, many of the same ordering constructs are equally applicable to the representation of the sequencing of states. OWL-S defines eight control constructs:

- Perform: execution of an action
- OrderedList: a list of control constructs to be done in order
- Split: a "bag" of process components to be executed concurrently. The Split completes when all of its component processes have been scheduled for execution.

- Split+Join: concurrent execution of a bunch of process components with synchronization. Split+Join completes when all of its components processes have completed.
- Any-Order: process components (specified as a bag) to be executed in some unspecified order but not concurrently. All components must be executed.
- Choice: the execution of a single control construct from a given bag of control constructs.
- If-Then-Else: intended as ``Test If-condition; if True do Then, if False do Else''
- Iterate: makes no assumption about how many iterations are made or when to initiate, terminate, or resume. The initiation, termination, or maintenance condition could be specified with a whileCondition or an untilCondition.
- Repeat-While/Repeat-Until: Both of these iterate until a condition becomes false or true. Repeat-While tests for the condition, exits if it is false and does the operation if the condition is true, then loops. Repeat-Until does the operation, tests for the condition, exits if it is true, and otherwise loops.

Table 2: Initial State Representation Ordering Constructs.

OWL-S Control Construct	Adapted State Representation Ordering Construct	State Representation Definition
Perform	Exists	A state relationship must exist
Sequence	OrderedList	A set of state relationships that must occur in a specific order
Any-Order	Any-Order	A set of state relationships that must all occur in any order
Iterate	Count	A state relationship that must be present multiple times.
n/a	NotExists	State relationship that can't exist.
Choice	Choice	A set of possible state relationships that can occur after a given state relationship
Join	Co-Exist	Two or more state relationships that must be true

We adapt some of the these control constructs to represent the ordering of states by changing their name and definition as shown in Table 2. Column 2 in Table 2 shows the state ordering constructs that we define in this work. This is not an exhaustive list, but the constructs shown should be sufficient for representing the kit assembly shown later.

Another interesting aspect not represented in OWL-S is the representation of state relationships that cannot occur for an intention to be true. For example, in the assembly of Kit 2 (shown later), we state that Part D is not a part of that kit. Thus, if we see a state in which Part D is placed within the Kit Tray, we can logically assume that Kit 2 is not the kit being assembled. Therefore, in addition to needing an 'Exists' construct, we also need a 'NotExists' construct, as shown in Table 2.

In the ontology, we represent the ordering constructs above as subclasses of the overall "OrderingConstructs" class. In addition, the state representations listed above are modeled as subclasses of the overall StateRepresentation class. States refer to an unordered list (a bag) of state relationships which completely describe the state. A state contains one to many state relationships. StateRelationships are defined using the RCC8 relationships described earlier. Table 3 shows what each construct points to and the cardinality restrictions.

Table 3: Construct Details

Construct	Points to:	Cardinality
State	StateRelationshipBag	1:n StateRelationships
StateRelationships	SolidObject	Exactly 2 SolidObjects
Ordering Constructs		
AnyOrder	OrderingConstructsBag	2:n OrderingConstructs
Choice	OrderingConstructsBag	2:n OrderingConstructs
CoExists	OrderingConstructsBag	2:n OrderingConstructs
Count	OrderingConstruct	Exactly 1 OrderingConstruct
Exists	StateRelationship	Exactly 1 StateRelationship
NotExists	StateRelationship	Exactly 1 StateRelationship
OrderedList	OrderingConstructsList	2:n OrderingConstructs

6. THE MANUFACTURING KITTING DOMAIN

Though we expect the approaches described in this chapter to be generic, we are initially applying them to a specific manufacturing domain to show their feasibility. In this domain, we focus on manufacturing kitting operations as described in (Balakirsky, 2012). Kitting is the process in which several different, but related items are placed into a container and supplied together as a single unit (kit) as shown in **Figure 6**. Kitting is often performed prior to final assembly in industrial assembly of manufactured products so all of the necessary parts are gathered in one location. Manufacturers utilize kitting due to its ability to provide cost savings, including saving manufacturing or assembly space, reducing assembly workers' walking and searching times, and increasing line flexibility and balance.

In batch kitting, the kit's component parts may be staged in containers positioned in the workstation or may arrive on a conveyor. Component parts may be fixtured, for example, placed in compartments on trays, or may be in random orientations, for example placed in a large bin. In addition to the kit's component parts, the workstation usually contains a storage area for empty kit boxes as well as completed kits.

Figure 6: Example Kit (courtesy of LittleMachineShop.com)

Kitting has not yet been automated in many industries where automation may be feasible.

Consequently, the cost of building kits is higher than it could be (Balakirsky et al., 2012). We are addressing this problem by building models of the knowledge that will be required to operate an automated kitting workstation in an agile manufacturing environment. For our automated kitting workstation, we assume that a robot performs a series of pick-and-place operations in order to construct the kit. These operations include:

1. Pick up empty kit and place on work table.
2. Pick up multiple component parts and place in kit.
3. Pick up completed kit and place in full kit storage area.

Each of these actions may be a compound action that includes other actions such as end-of-arm tool changes, path planning, and obstacle avoidance. Finished kits are moved to the assembly floor where components are picked from the kit for use in the assembly procedure. The kits are normally designed to facilitate component picking in the correct sequence for assembly. Component orientation may be constrained by the kit design in order to ease the pick-to-assembly process. Empty kits are returned to the kit building area for reuse.

7. A KITTING ONTOLOGY

In late 2011, IEEE formed a working group entitled Ontologies for Robotics and Automation (ORA) (Schlenoff, 2012b). The goal of the working group is to develop a standard ontology and associated methodology for knowledge representation and reasoning in robotics and automation, together with the representation of concepts in an initial set of application domains. The working group understood that it would be extremely difficult to develop an ontology that could cover the entire space of robotics and automation. As such, the working group is structured in such a way as to take a bottom-up and top-down approach to addressing this broad domain. This group is comprised of four sub-groups entitled: Upper Ontology/Methodology (UpOM), Autonomous Robots (AuR), Service Robots (SeR), and Industrial Robots (InR). The InR, AuR, and SeR sub-groups are producing sub-domain ontologies that will serve as a test case to validate the upper ontology and the methodology developed by UpOM.

The industrial robots group is focusing on manufacturing kitting operations as a test case. This kitting ontology is focusing on activities that are expected to be performed in a sample kitting operation along with pertinent objects that are expected to be present.

7.1. REPRESENTING KITTING OBJECTS IN THE ONTOLOGY

The industrial kitting objects ontology is written in Web Ontology Language (OWL)(Harmelen, 2004). Conceptually, the model is an object model in which there are classes with attributes (but no functions, constructors, etc.). OWL has classes but does not have attributes; it has ObjectProperties and DataProperties instead. They may be used to model attributes. OWL Properties are global, not local to a class, so localizing each attribute to a class is done by a naming convention that includes the class name and the attribute name. OWL supports multiple inheritance, but that has not been used in the kitting ontology. Except by subclass relationship, no object is in more than one class.

Table 4: Subset of the Kitting Object Ontology

SolidObject *PrimaryLocation SecondaryLocation*
BoxyObject *Length Width Height*
KitTray *SkuRef SerialNumber*
LargeContainer *SkuRef SerialNumber*
PartsBin *PartQuantity PartSkuRef*
PartsTray *SkuRef SerialNumber*
Kit *DesignRef Tray Parts*
KittingWorkstation *WorkTable Robot ChangingStation OtherObstacles AngleUnit LengthUnit WeightUnit Skus KitDesigns*
LargeBoxWithEmptyKitTrays *LargeContainer Trays*
LargeBoxWithKits *LargeContainer Kits Capacity*
Part *SkuRef SerialNumber*
PartsTray *SkuRef SerialNumber*
PartsTrayWithParts *Tray Parts*
Robot *Description Id WorkVolume EndEffector MaximumLoadWeight*
WorkTable *SolidObjects*
DataThing
PhysicalLocation *RefObject*
PoseLocation *Point ZAxis XAxis*
PoseLocationIn
PoseLocationOn
PoseOnlyLocation
RelativeLocation
RelativeLocationIn
RelativeLocationOn

The model has two top-level classes, **SolidObject** and **DataThing**, from which all other classes are derived. **SolidObject** models solid objects, things made of matter. **DataThing** models data. Subclasses of **SolidObject** and **DataThing** are defined as shown in Table 4. The level of indentation indicates subclassing. Items in *italics* following class are names of class attributes. Derived types inherit the attributes of the parent. An attribute with a solid underline must occur at least once and can also occur many times. An attribute with a dashed underline may occur zero, one, or many times. An attribute with a dotted underline may occur once or not at all. An attribute with no underline must occur exactly once. Each attribute has a specific type not shown in Table 4. If an attribute type has derived types, any of the derived types may be used.

Each **SolidObject** has a native coordinate system conceptually fixed to the object. The native coordinate system of a **BoxyObject**, for example, has its origin at the middle of the bottom of the object, its Z axis perpendicular to the bottom, and the X axis parallel to the longer horizontal edges of the object.

Each **SolidObject A** has at least one **PhysicalLocation** (the *PrimaryLocation*). A **PhysicalLocation** is defined by giving a reference **SolidObject B** and information saying how the position of **A** is related to **B**. Two types of location are required for operation of the kitting workstation. Relative locations,

specifically the knowledge that one **SolidObject** is in or on another, are needed to support making logical plans for building kits. Mathematically precise locations are needed to support robot motion. The mathematical location, **PoseLocation**, gives the pose of the coordinate system of **A** in the coordinate system of **B**. The mathematical information consists of the location of the origin of **A**'s coordinate system and the directions of its Z and X axes. The mathematical location variety has subclasses representing that, in addition, **A** is in **B (PoseLocationIn)** or on **B (PoseLocationOn)**. The subclasses of **RelativeLocation** are needed not only for logical planning, but also for cases when the relative location is known, but the mathematical information is not available. This occurs, for example when a **PartsBin** is being used, since by definition, the **Parts** in a **PartsBin** are located randomly.

All chains of location from **SolidObject**s to reference **SolidObject**s must end at a **KittingWorkstation** (which is the only class of **SolidObject** allowed to be located relative to itself).

The kitting ontology includes several subclasses of **SolidObject** that are formed from components that are **SolidObject**s. These are: **Kit**, **PartsTrayWithParts**, **LargeBoxWithEmptyKitTrays**, and **LargeBoxWithKits**. Combined objects may come into existence or go out of existence dynamically when a kitting workstation is operating. For example, when all the parts in a **PartsTrayWithParts** have been removed and put into kits, the **PartsTrayWithParts** should go out of existence and the **PartsTray** that was holding **Part**s should have its location switched from its location relative to the **PartsTrayWithParts** to the former location of the **PartsTrayWithParts**.

In the current version of the kitting ontology, the only way a shape can be described is by using a **BoxyObject**. This is adequate for making kits of boxes, but is not adequate for most industrial forms of kitting. The kitting ontology includes **ShapeDesign** as a stub, but its only attribute is a string giving a description of the shape. For manipulating non-boxy **SolidObject**s some mathematically usable representation of shape will be required.

7.2. REPRESENTING ACTIVITIES IN THE ONTOLOGY

To represent activities in the manufacturing kitting ontology, both the actions and the pre- and post-conditions of those actions need to be represented. OWL-S (Martin et al., 2004) is used to represent the actions that need to be performed and SWRL atoms are used to represent the preconditions and effects of the actions. Preconditions and effects are represented as state relationships. An example of the take-kt (take kit tray) action is shown in Equation 18. In natural language, the take-kt action involves a robot (*r*) equipped with an end effector (*eff*) picking up a kit tray (*kt*) from within a large box with empty kit trays (*lbwekt*). The action is formally defined in the State Variable Representation (Nau, 2004) as:

take-kt(*r, kt, lbwekt, eff, wtable*) (18)

Figure 7 shows the preconditions and effects that are associated with this action.

Figure 7: Preconditions and Effects for the Take Kit Tray Action

Each of the state relationships in the table above is described below:

1. rhold-empty(*r*) – TRUE iff robot (*r*) is not holding anything
2. lbwekt-non-empty(*lbwekt*) – TRUE iff Large Box With Empty Kit Trays (*lbwekt*) is not empty
3. r-with-eff(*r, eff*) – TRUE iff Robot (*r*) is equipped with an EndEffector (*eff*)

4. ktlocation(*kt, lbwekt*) – TRUE iff the Kit Tray (*kt*) is in the Large Box With Empty Kit Trays (*lbwekt*)
5. efflocation(*eff*, *r*) – TRUE iff the EndEffector (*eff*) is being held by the robot (*r*)
6. worktable-empty(*wtable*) – TRUE iff there is nothing on the Worktable (*wtable*)
7. efftype(*eff, kt*) – TRUE iff the EndEffector (*eff*) is designed to handle the Kit Tray (*kt*)
8. ¬rhold-empty(*r*) – TRUE iff the robot (*r*) is holding something
9. ktlocation(*kt, r*) – TRUE iff the Kit Tray (kt) is being held by the Robot (*r*)
10. rhold(*r, kt*) – TRUE iff the Robot (*r*) is holding the Kit Tray (*kt*)
11. ¬ktlocation(*kt, lbwekt*) – TRUE iff the Kit Tray (*kt*) is not in the Large Box With Empty Kit Trays (*lbwekt*)

There are many other actions that can be performed during the kitting operation, including putting down a kit tray, picking up and putting down a part, attaching/removing an end effector, etc. Each of these actions has associated preconditions and effects. Sections 10, 11, and 12 will show a novel approach to how the truth-value of these pre- and post-conditions (state relationships) can be inferred using the output of object recognition and pose recognition algorithms from a sensor system.

8. STATE REPRESENTATION IN THE MANUFACTURING KITTING DOMAIN

When modeling the state relationships (preconditions and effects) shown in Section 7.2 above, the first step is to precisely define the state relationships in such a way as to determine if there were similar spatial relations that could be leveraged. We can start to formalize the previous definition of the state relationships as follows:

The rholds state relationships (1, 8, and 10) depend on the type of effector that is being used to define the state relationship. We will assume there are two types of end effectors: a vacuum end effector and a parallel gripper end effector. The vacuum end effector picks objects up by positioning itself on top of the object and uses air to create a vacuum to adhere to the object. The parallel gripper end effector picks objects up by squeezing them from both sides. Because the vacuum end effector would not reasonably be used to pick up the kit tray, the vacuum-rhold(*r, kt*) state is not included below. In the case of the vacuum end effector, the relevant state relationships would be:

- vacuum-rhold-empty(*r*) – there is no object **under and in contact with** the robot vacuum effector
- ¬vacuum-rhold-empty(*r*) – there is an object **under and in contact with** the robot vacuum effector

In the case of the parallel gripper end effector, the state relationships would be

- gripper-rhold-empty(*r*) – there is no object **partially in and in contact with** the robot gripper
- ¬gripper-rhold-empty(*r*) – there is an object **partially in and in contact with** the robot gripper
- gripper-rhold(*r, kt*) – the Kit Tray is **partially in and in contact with** the robot gripper

Table 5: Revised definitions of spatial relationships

# from Above	State Relationship	Previous Definition	Revised Definition
(2)	lbwekt-non-empty(*lbwekt*)	TRUE iff an object is in the Large Box With Empty Kit Trays (*lbwekt*)	there is an object that is **contained in** the Large Box With Empty Kit Tray
(3)	r-with-eff(*r, eff*)	TRUE iff Robot (*r*) is equipped with an EndEffector (*eff*)	the effector is **in contact with** the robot
(4)	ktlocation(*kt, lbwekt*)	TRUE iff the Kit Tray (*kt*) is in the Large Box With Empty Kit Trays (*lbwekt*)	the Kit Tray is **contained in** the Large Box With Empty Kit Trays
(5)	efflocation(*eff*, *r*)	TRUE iff the EndEffector (*eff*) is being held by the robot (*r*)	the effector is **in contact with** the robot (Note: Same as (3) above)
(6)	worktable-empty(*wtable*)	TRUE iff there is nothing on the Worktable (*wtable)*	there is no object that is **on top of** and **in contact with** the Worktable
(7)	efftype(*eff, kt*)	TRUE iff the EndEffector (*eff*) is designed to handle the Kit Tray (*kt*)	the Effector can handle the Kit Tray
(9)	ktlocation(*kt, r*)	TRUE iff the Kit Tray (kt) is being held by the Robot (*r*)	the Kit Tray is **in contact with** the Robot and there is nothing **under and in contact with** the Kit Tray
(11)	¬ktlocation(*kt, lbwekt*)	TRUE iff the Kit Tray (*kt*) is not in the Large Box With Empty Kit Trays (*lbwekt*)	the Kit Tray is **not contained in** the Large Box With Empty Kit Trays

There is also one state relationship that does not rely on spatial relations. The definition of efftype(*eff, kt*) states that a specific effector must be able to be used on a kit tray. It is envisioned that this information will be included in the ontology class to describe the kit tray and therefore is out of scope of this document.

Also, in the manufacturing kitting domain, not all objects can be represented as convex regions, as is required by the RCC8 formalism. For example, the robot gripper in Figure 8 is not convex and thus does not neatly fit into the RCC8 approach. To address this, we develop a convex hull along each relevant plane (as shown in Figure 8) around objects of this sort and use that convex hull to represent the region of the object in that plane.

Figure 8: Convex Hull Around Robot Gripper

Based on the manufacturing kitting ontology described in Section 7.1. and the spatial relations described in Section 4, we can formally define the 11 manufacturing kitting state relationships:

$$\text{lbwekt-non-empty}(lbwekt) \rightarrow \text{SolidObject}(obj1) \wedge \text{Contained-In}(obj1, lbwekt) \tag{19}$$

$$\text{r-with-eff}(r, eff) \rightarrow \text{In-Contact-With}(r, eff) \tag{20}$$

$$\text{ktlocation}(kt, lbwekt) \rightarrow \text{Contained-In}(kt, lbwekt) \tag{21}$$

$$\text{worktable-empty}(wtable) \rightarrow \text{SolidObject}(obj1) \wedge \neg\text{On-Top-Of}(obj1, wtable) \wedge \neg\text{In-Contact-With}(obj1, wtable) \tag{22}$$

$$\text{gripper-rhold}(r, kt) \rightarrow \text{GripperEffector}(eff) \wedge \text{r-with-eff}(r, eff) \wedge \text{Partially-In-And-In-Contact-With}(kt, eff) \tag{23}$$

$$\neg\text{ktlocation}(kt, lbwekt) \rightarrow \neg\text{Contained-In}(kt, lbwekt) \tag{24}$$

$$\text{vacuum-rhold-empty}(r) \rightarrow \text{SolidObject}(obj1) \wedge \text{SolidObject}(obj2) \wedge \text{VacuumEffector}(eff) \wedge \text{r-with-eff}(r, eff) \wedge \neg(\text{Under-And-In-Contact-With}(obj1, eff) \wedge \neg\text{Under-And-In-Contact-With}(obj2, obj1)) \tag{25}$$

$$\neg\text{vacuum-rhold-empty}(r) \rightarrow \text{SolidObject}(obj1) \wedge \text{SolidObject}(obj2) \wedge \text{VacuumEffector}(eff) \wedge \text{r-with-eff}(r, eff) \wedge \text{Under-And-In-Contact-With}(obj1, eff) \wedge \neg\text{Under-And-In-Contact-With}(obj2, obj1) \tag{26}$$

$$\text{gripper-rhold-empty}(r) \rightarrow \text{SolidObject}(obj1) \wedge \text{GripperEffector}(eff) \wedge \text{r-with-eff}(r, eff) \wedge \neg\text{Partially-In-And-In-Contact-With}(obj1, eff) \tag{27}$$

$$\neg \text{gripper-rhold-empty}(r) \rightarrow \text{SolidObject}(obj1) \wedge \text{GripperEffector}(eff) \wedge \text{r-with-eff}(r, eff) \wedge \text{Partially-In-And-In-Contact-With}(obj1, eff) \quad (28)$$

$$\text{ktlocation}(kt, r) \rightarrow \text{gripper-rhold}(r, kt) \quad (29)$$

The formal definitions of these high-level state relationships will allow their existence to be recognized in a manufacturing environment, which in turn can be used by a state-based intention recognition system. The presence of states in certain predefined orders can help a robot recognize the intention of a human in the environment, which would allow the robot to better assist the human in performing upcoming activities. In the following detailed, yet simple, example, we will show how these state relationships can be represented, sequenced, and used to infer the intention of the performer (human).

9. KITTING EXAMPLE

In this section, we will use the detailed scenario shown in Figure 9. In this scenario, a person (not shown) is constructing one of two possible kits. Both kits use the same kit tray (Kit Tray) and contain a series of parts placed in the kit in any order. Kit 1 contains two Part A's, two part B's, one Part C, and one Part D. Kit 2 contains three part A's, one part B, and one part C. This is shown in Figure 10. In this scenario, all parts with the same letter are identical (e.g., all A's are the same, all B's are the same). Other parts are available at the workstation and could be used for other kit assemblies that are not of interest for this example. A table is provided on which all work is performed. Parts and part trays are available from a set of boxes that can be refilled as needed. When a kit is completed, it is placed in a completed kit box (not shown).

Figure 9: Sample Kitting Scenario

Figure 10: Completed Kits 1 and 2

A person is tasked with creating these kits. The person may choose to create either of these kits at any given time. A robot is available to assist the person by inferring which kit the person is intending to create at a given time and providing support where needed. The goal of the robot is to infer the intention of the human based on what it perceives (i.e., which kit tray the human is assembling).

10. REPRESENTING STATES, STATE RELATIONSHIPS, AND INTENTIONS IN AN ONTOLOGY

For Kit 1, we can infer from the description above that the process for assembling this kit would initially involve placing the kit tray (Kit Tray) on the table and then adding, in any order, two Part A's, two Part B's, one Part C, and one Part D. Similarly, for Kit 2, we can deduce from the description above that for the process for assembling this kit would initially involve placing the kit tray (Kit Tray 1) on the table and then adding, in any order, three Part A's, one Part B, and two Part C's. We can now use the state relations described in Section 4 and the ordering relationships in Section 5 to build intentions.

For Kit 1, we start by requiring that the Kit Tray be on the Table (where SR stands for state relation and OC stands for Ordering Construct):

$$SR1 = \text{On-Top-Of}(\text{KitTray}, \text{Table}) \tag{30}$$

$$OC1 = \text{Exists}(SR1) \tag{31}$$

Next, we represent all of the states that can happen in any order:

$$SR2 = \text{Contained-In}(\text{PartA}, \text{KitTray}) \tag{32}$$

$$SR3 = \text{Contained-In}(\text{PartB}, \text{KitTray}) \tag{33}$$

$$SR4 = \text{Contained-In}(\text{PartC}, \text{KitTray}) \tag{34}$$

$$SR5 = \text{Contained-In}(\text{PartD}, \text{KitTray}) \tag{35}$$

$$OC2 = \text{Count}(SR2, 2) \tag{36}$$

$$OC3 = \text{Count}(SR3, 2) \tag{37}$$

$$OC4 = \text{Count}(SR4, 1) \tag{38}$$

$$OC5 = \text{Count}(SR5, 1) \tag{39}$$

$$OC6 = \text{AnyOrder}(OC2, OC3, OC4, OC5) \tag{40}$$

$$SR6 = \text{Contained-In}(\text{KitTray}, \text{CompletedKitBox}) \tag{41}$$

$$OC7 = \text{Exists}(SR6) \tag{42}$$

We also need to represent the states that cannot be true if this kit assembly is indeed the intention. In reality, there are likely an infinite number of states that could make this intention be deemed false. However, for this work, we will specifically focus on the states that are possible in other intentions but are not possible here. Some of these states may include:

$$SR7 = \text{Contained-In}(\text{PartE}, \text{KitTray}) \tag{43}$$

$$OC8 = \neg \text{Exists}(SR7) \tag{44}$$

Based on the above, we can represent the intention of the Kit 1 assembly as follows:

$$OC9 = OrderedList(OC1, OC6, OC7) \ \& \ OC8 \quad (45)$$

The & symbol in Equation 45 represents that the ordering constraint after the & symbol (OC8 in this case) must be true throughout the entire process (OC9).

Using OC9 above, and applying the same approach to the Kit 2 intention (not shown), we can use a template-based approach to represent the sequence associated with Kit 1 and Kit 2 intention, which may look something like what is shown in Table 6 and Table 7:

Table 6: Kit 1 Template-Based Intention Sequences

State Relation	**1**	**2**	**2**	**3**	**3**	**4**	**5**	**6**	**7**
Kit 1	On_Top_Of (KitTray, Table)	Contained-In (PartA, KitTray)	Contained-In (PartA, KitTray)	Contained-In (PartB, KitTray)	Contained-In (PartB, KitTray)	Contained-In (PartC, KitTray)	Contained-In (PartD, KitTray)	Contained-In (KitTray, CompletedKitBox)	Not(Contained-In(PartE, KitTray))
Previous State Relation	n/a	1	1	1	1	1	1	2, 3, 4, 5	n/a

Table 7: Kit 2 Template-Based Intention Sequence

State Relation	**1**	**2**	**2**	**2**	**3**	**4**	**5**	**6**	**7**
Kit 2	On_Top_Of (KitTray, Table)	Contained-In (PartA, KitTray)	Contained-In (PartA, KitTray)	Contained-In (PartA, KitTray)	Contained-In (PartB, KitTray)	Contained-In (PartC, KitTray)	Contained-In (KitTray, CompletedKitBox)	Not(Contained-In(PartD, KitTray	Not(Contained-In(PartE, KitTray))
Previous State Relation	n/a	1	1	1	1	1	2, 3, 4	n/a	n/a

All state relationships are listed, and the ones that must be true more than once (as represented by the Count construct in the ontology) are represented in separate columns. The state relationships that cannot be true for a specific intention are represented at the end of the table. Under each state relationship is a pointer to the state relationship that must occur before this one is relevant. For example, before the Contained-In(PartA, KitTray) state relationship can be evaluated as being true, the On-Top_Of(KitTray, Table) (as indicated by the number 1 at the top of the table) must have occurred. This table will be used in subsequent sections to show how intentions can be recognized based upon observations.

11. TRACKING STATES AND STATE TRANSITIONS AS ACTIONS OCCUR

Now that we have specified how the states, state relationships, and intentions are represented in the ontology, we will show how state relationships are tracked by observations and associated with the intentions. Based on the detailed kitting example described in Section 9, the first step is to determine which objects (and spatial relations) are relevant to be tracked. In this domain, the relevant objects are:

- Part A
- Part B
- Part C
- Part D
- Part E
- Kit Tray
- Table
- Part A Box
- Part B Box
- Part C Box
- Part D Box
- Completed Kit Box

Note that Part E is of interest not because it is a part if the kit assembly, but because it is explicitly prohibited from being in one of the kit assemblies. Even though Part F is available within the environment, it is not of interest to either intention and is therefore not tracked. Using the approaches described above, the robot first extracts the state relationships that are relevant to the various intentions it is trying to perceive. These state relations are described in the previous section. In the cases of the two kit assemblies, the relevant state relationships include:

- Part A is in Part A Box
- Part B is in Part B Box
- Part C is in Part C Box
- Part D is in Part D Box
- KitTray is in Large Box With Empty Kit Trays
- KitTray is on Table
- Part A is in the KitTray
- Part B is in the KitTray
- Part C is in the KitTray
- Part D is in the KitTray
- Part D is not in the KitTray
- Part E is not in the KitTray
- Kit Tray is in Completed Kit Box

The truth-value of these state relationships can be evaluated at a given point in time by applying the state representation approach based on RCC8 as described in the previous section. In tabular format, the state relationships can be represented as shown in **Table 8**. Actions occur between each state to cause the truth-value of the relevant state relations to change. However, for this approach, all we care about is the truth-value of the state relations without needing to track the actions that cause them to be true or false.

State 1 shows the state of the environment as described in Figure 9. Numbers in the cells represent how many instances of the state relationship are true. In this example, there are three instances of Part A in Part A Box, as shown in the figure. If, at the next state, the Kit Tray was removed from the Kit Tray Holder and placed on the Table, the next state would be represented as in State 2. If one of the Part A's is then removed from the Part A Box and placed into the Kit Tray, the state representation would look like State 3.

Table 8: Tabular State and State Relationship Representation.

State Relation	State 1 (Initial State)	State 2 (Kit Tray on Table)	State 3 (Part A in Kit Tray)	…
Part A is in Part A Box	3	3	2	
Part B is in Part B Box	2	2	2	
Part C is in Part C Box	1	1	1	
Part D is in Part D Box	1	1	1	
Kit Tray is in Large Box With Empty Kit Trays	1	0	0	
Kit Tray is on Table	0	1	1	
Part A is in Kit Tray	0	0	1	
Part B is in Kit Tray	0	0	0	
Part C is in Kit Tray	0	0	0	
Part D is in Kit Tray	0	0	0	
Part E is in Kit Tray	0	0	0	
Kit Tray in Completed Kit Box	0	0	0	

Understanding the states that are relevant also plays a significant role in determining which sensors to use and where to place the sensors that are meant to track objects in the environment. For example, if we assume that Part Box A is not a clear box and there is an opening at the top, a sensor would need to be placed above the box so that it is evident if (and how many) objects are present within the box (addressing the state relationship "Part A is in Part A Box").

In this approach, object locations are tracked in real-time by sensors. However, the state relationships shown in **Table 8** are only evaluated when a new state is reached. States are defined as a period of time in which an object of interest (as described above) has moved greater than a predefined distance from its previous position and has stopped moving for greater than a predefined period of time. The distance and time values are up to the user, and are often a function of the domain. For an operation where objects are relatively close together (such as a scenario where all objects are already on a table), the distance metric will likely be very small. In other domains where objects are moved greater distances

(such as a large assembly operation), this distance metric will likely be much greater. States can happen over various periods of time. In the table above, State 1 is meant to represent the initial state of the system. State 2 may be achieved a few seconds after the observations begin, while State 3 may not be achieved until minutes or hours afterwards. Many objects may have changed state between State 1 and State 2, but in this case, those objects of interest and their corresponding state relationships of interest have not changed.

12. MATCHING PERCEIVED STATES TRANSITIONS TO ONTOLOGY INTENTION REPRESENTATIONS

As mentioned above, a new state becomes true when a state relationship that is being tracked changes its truth value. As an example, if the Kit Tray goes from not being on the table to being on the table, a new state is formed. During each state, the state relationships that are true are compared to the intention templates shown in Table 6. However, only the state relationships in Table 6 that are eligible are available for matching.

A state relationship becomes eligible if all required previous state relationships have occurred. For example, in Table 6 for Kit 1, the state relationship Contained-In(PartA, KitTray) can not occur until On_Top_Of (Kit Tray, Table) has occurred. Therefore, even if Contained-In(PartA, KitTray) is true in the observed scene, it is not active unless On_Top_Of (Kit Tray, Table) has previously occurred and thus should not be "checked off" unless that previous condition is met.

As each observed state occurs, the corresponding state relationships are compared with those associated with the possible intentions. When the observed state relationship(s) match an active state relationship in an intention, it is "checked off" indicating that the state relationship has occurred. This process occurs with every new observed state that occurs. In addition, after each observed state and associated matching with the intentions, the number of state relations that are true for that intention are summed. Continuing with our example, if the state table was updated to look like what is shown in Table 9, the corresponding intention table would look like what is shown in Table 10 and Table 11, with green cells representing those state relationships which have been observed and whose precondition state relations have been met.

Table 9: Tabular State and State Relationship Representation (Version 2)

State Relation	State 1 (Initial State)	State 2 (Kit Tray On Table)	State 3 (Part A In Kit Tray)	State 3 (2^{nd} Part A In Kit Tray)	State 4 (Part B in Kit Tray)	State 5 (Part C In Kit Tray)
Part A is in Part A Box	3	3	2	1	1	1
Part B is in Part B Box	2	2	2	2	1	1
Part C is in Part C Box	1	1	1	1	1	0
Part D is in Part D Box	1	1	1	1	1	1
Kit Tray is in Large Box With Empty Kit Trays	1	0	0	0	0	0
Kit Tray is on Table	0	1	1	1	1	1
Part A is in Kit Tray	0	0	1	2	2	2
Part B is in Kit Tray	0	0	0	0	1	1
Part C is in Kit Tray	0	0	0	0	0	1
Part D is in Kit Tray	0	0	0	0	0	0
Part E is in Kit Tray	0	0	0	0	0	0
Kit Tray in Completed Kit Box	0	0	0	0	0	0

Table 10: Matching Observed States to the Kit 1 Intention

State Relation	1	2	2	3	3	4	5	6	7	Sum
Kit 1	On_Top_Of (Kit Tray, Table)	Contained-In (PartA, KitTray)	Contained-In (PartA, KitTray)	Contained-In (PartB, KitTray)	Contained-In (PartB, KitTray)	Contained-In (PartC, KitTray)	Contained-In (PartD, KitTray)	Contained-In (KitTray, CompletedKitBox)	Not(Contained-In(PartE, KitTray	5
Previous State Relations	n/a	1	1	1	1	1	1	2, 3, 4, 5	n/a	

Table 11: Matching Observations to the Kit 2 Intention

State Relation	1	2	2	2	3	4	5	6	7	Sum
Kit 2	On_Top_Of (Kit Tray, Table)	Contained-In (PartA, KitTray)	Contained-In (PartA, KitTray)	Contained-In (PartA, KitTray)	Contained-In (PartB, KitTray)	Contained-In (PartC, KitTray)	Contained-In (KitTray, CompletedKitBox)	Not(Contained-In(PartD, KitTray	Not(Contained-In(PartE, KitTray	5
Previous State Relations	n/a	1	1	1	1	1	2, 3, 4	n/a	n/a	

In this simple example, at a given state, the Kit 1 intention has five state relations that are matched and the Kit 2 intention also has five state relations that are matched. To determine which intention is more probable, we use the following equation:

$$P_{i,t} = \frac{S_{i,t}}{\sum_{i=1}^{n} S_{i,t}} \tag{46}$$

where $P_{i,t}$ represents the probability of intention *i* at time *t*, $S_{i,t}$ represents the sum of the state relationships that are true for intention *i* at time *t* and *n* represents the number of intentions that are being evaluated. So in this case, for the Kit 1 and Kit 2 intentions, the ratio would be 5/10 = 0.5 (or 50%).

In addition, we need to handle state relationships that cannot occur in an intention and adjust the probabilities accordingly. We modify equation (46) by adding a coefficient (F_t for forbidden at time *t*) in front of the equation, so it becomes:

$$P_{i,t} = F_{i,t} \frac{S_{i,t}}{\sum_{i=1}^{n} S_{i,t}} \tag{47}$$

F_t has a value of 0 or 1 depending if any observed state relationships are explicitly prohibited in the intention at the given time. F_t is initially set to 1. If a prohibited state relationship is observed, the value of F_t is set to 0 and the overall probably of the intention becomes 0. If no such state relationships exist, the value of F_t remains at 1.

So in Table 10, if we have a subsequent observation where Part D is placed into the kit tray, the Kit 1 intention would have six state relationships which are true and the Kit 2 intention would have five true relationship and one prohibit relationship. The number of true state relationships in the Kit 2 intention would drop to zero because of the prohibited state relationship. Thus, the probability of the Kit 1 intention would be 6/6 = 1 (100 %) and the probability of Kit 2 would be 0/6 = 0 (0 %).

It is important to note that prohibited state relations are treated differently than state relations that occur but do not advance the intention. For example, if an additional Part A is placed inside the kit, it would "advance" the Kit 2 intention since three Part A's are required for it. However, a third Part A is not explicitly prohibited for Kit 1 as it is for Part D. Therefore, the probability that Kit 2 is the intention would increase but the probability for Kit 1would not drop to zero since the addition of an third Part A is not explicitly forbidden.

One of the assumptions in many intention recognition systems is that there is a closed world. In other words, only the intentions that are being compared can happen and no other intentions can occur. In this work, we relax that assumption by ensuring that the intentions are progressing with each state observation. If we find that no tracked intention has increased the number of state relationships that are associated with it after a state observation, we lower the probability of all intentions and correspondingly increase the probability of an unknown intention category. In essence, we are saying that there is some intention being observed, but it does not match the intentions that we know about.

We capture this using a progress factor (PF), where the equation is shown below:

$$\begin{cases} if\ (S_{i,t} - S_{i,t-1}) > 0\ then\ PF_{i,t} = 1 \\ if\ (S_{i,t} - S_{i,t-1}) = 0\ then\ PF_{i,t} = 0.8 * PF_{i,t-1} \\ if\ (S_{i,t} - S_{i,t-1}) < 0\ then\ PF_{i,t} = 0.6 * PF_{i,t-1} \end{cases} \tag{48}$$

where $S_{i,t}$ represents the number of state relationships that are true at time *t* for intention *i* and $S_{i,t-1}$ represents the number of state relationships that are true at time *t-1* for intention *i*. If at any time $PF_{i,t} > 1$, then it is set to 1.

The values of 0.6 and 0.8 allow for degradation in the confidence that the intention is occurring. These values can be set by the user to be any values that are deemed appropriate for the domain of interest. At this point, the probability of the i^{th} intention would be:

$$(P_{i,t})_{temp} = PF_{i,t}F_{i,t}\frac{S_{i,t}}{\sum_{i=1}^{n} S_{i,t}} \tag{49}$$

The reason for the *temp* after the $P_{i,t}$ is because of the probability adjustment; the sum of the probabilities does not equal 100% and needs to be normalized. This can happen in one of two fashions. In the case where at least one intention retains a PF of 1 (in other words, at least one intention progresses since the last state), all probabilities are normalized to result in a total probability of 100 %, as shown below

$$P_{i,t} = \frac{(P_{i,t})_{temp}}{\sum_{1}^{n}(P_{i,t})_{temp}} \tag{50}$$

where n is the total number of intentions being matched.

Table 12: Tabular State and State Relationship Representation (Version 2)

State Relation	State 1 (Initial State)	State 2 (Kit Tray On Table)	State 3 (Part A In Kit Tray)	State 3 (2nd Part A In Kit Tray)	State 4 (Part B in Kit Tray)	State 5 (Part C In Kit Tray)	State 6 (Part A Removed From Part A Box)
Part A is in Part A Box	3	3	2	1	1	1	0
Part B is in Part B Box	2	2	2	2	1	1	1
Part C is in Part C Box	1	1	1	1	1	0	0
Part D is in Part D Box	1	1	1	1	1	1	1
Kit Tray is in Large Box With Empty Kit Trays	1	0	0	0	0	0	0
Kit Tray is on Table	0	1	1	1	1	1	1
Part A is in Kit Tray	0	0	1	2	2	2	2
Part B is in Kit Tray	0	0	0	0	1	1	1
Part C is in Kit Tray	0	0	0	0	0	1	1
Part D is in Kit Tray	0	0	0	0	0	0	0
Part E is in Kit Tray	0	0	0	0	0	0	0
Kit Tray in Completed Kit Box	0	0	0	0	0	0	0

In the case where no intentions retain a PF of 1 (in other words, no intention progresses since the last state), we assume that something is occurring in the environment that does not correspond to one of the

predefined intentions that we are exploring. In this case, we continue to lower the probability of each known intention as shown in Equation 49, and assign the remaining probability that was deducted from the known intentions to a new "unknown" intention. As more time passes and no intention continues to be matched, the known intentions continue to get less probable and the unknown intention becomes more probable.

In Table 12, a new state has occurred in which a Part A was removed from the Part A Box, but not placed in either of the existing kit trays. Because this is a relevant state relationship, it is recorded in the state table but does not directly impact the two intentions we are tracking. As such, the intention table (Table 10) does not change. What this implies is that a new state occurred, but did not advance either of the two intentions. In this example, the probability of the Kit 1 intention becomes 0.9 ((progress factor) * 1 (forbidden state relations) * 5 (true state relationships)) / 10 (total true state relationships for all intentions) = 0.45 or 45 %. Similarly for the Kit 2 intention, the probability becomes 45 %. The remaining 10 % is assigned to the unknown intention, since no intention progressed, thus implying that there is an intention going on in the environment that we are unaware of.

13. CONCLUSION

In this chapter, we present an approach to representing state information as the basis for intention recognition and then show how these states and their sequences can be used to associate probabilities with various intentions. The state representation is based on RCC8 and various cardinal directions along the x-, y-, and z- axes. Intention recognition is based upon a form of template matching that incorporates sequence information and state relationships that cannot occur. A progress metric is also introduced to decrease the probability that an intention is occurring if no progress has been made on it in recent states, which also allows for an open world assumption that intentions could be occurring in the environment that are not previously known. A small set of abstract state relationships is defined (e.g., under, on-top-of, contained-in) as the basis for defining domain-specific state relationships (e.g., worktable-empty, ktlocation, etc.). A simple example of two kit intentions is included to explain each step in the process.

State-based intention recognition offers some interesting advantages over activity-based recognition (which has traditionally been performed in the literature), including:

- States are often more easily recognizable by sensor systems than actions, as shown in Section 3.
- Using activities, intention recognition is often limited to inferring the intention of a single person. State-based intention recognition eliminates this shortfall, in that the state relationship is independent of who created it.
- State information is often more ubiquitous than activity information, thus allowing for reusability of the ontology.

Because of the similarity of state representation with activity representation, many of the same approaches that were described in the "Intention Recognition and State Representation Related Work" section can also be applied to this approach, which will also be the subject of future work. Additional future work will explore the association of weights with various state relationships. For example, if a state relationship exists in the environment that is only relevant to one intention, that state relationship should receive a higher weight since it is a strong indicator that the associated intention is occurring.

DISCLAIMER

Certain commercial software and tools are identified in this chapter in order to explain our research. Such identification does not imply recommendation or endorsement by the National Institute of Standards and Technology, nor does it imply that the software tools identified are necessarily the best available for the purpose.

REFERENCES

Albath, J., Leopold, J., Sabharwal, C., & Maglia, A. (2010). *RCC-3D: Qualitative Spatial Reasoning in 3D*. Paper presented at the 23rd International Conference on Computer Applications in Industry and Engineering (CAINE), Las Vegas, NV.

Balakirsky, S., Kootbally, Z., Schlenoff, C., Kramer, T., & Gupta, S. (2012). *An Industrial Robotic Knowledge Representation for Kit Building Applications*. Paper presented at the International Robots and Systems (IROS) Conference Vilamoura, Algarve Portugal.

Bateman, J., & Farrar, S. (2006). Spatial Ontology Baseline Version 2.0 *OntoSpace Project Report - Spatial Cognition SFB/TR 8: I1 - [OntoSpace]*: University of Bremen.

Bodor, R., Jackson, B., & Papanikolopoulos, N. (2003). *Vision-Based Human Tracking and Activity Recognition*. Paper presented at the 11th Mediterranean Conference on Control and Automation.

CCC. (2009). A Roadmap for US Robotics: From Internet to Robotics http://www.us-robotics.us/reports/CCC Report.pdf: **Computing Community Consortium.**

Chabrol, J. (1987). Industrial robot standardization at ISO. *Robotics, 3*(2).

Choudhury, T., & Borriello, G. (2008). The Mobile Sensing Platform: An Embedded System for Activity Recognition. *IEEE Pervasive Magazine- Special Issue on Activity-Based Computing*.

DARPA. (2012). Information Innovation Office: Mind's Eye Program http://www.darpa.mil/Our_Work/I2O/Programs/Minds_Eye.aspx**.**

Demolombe, R., Mara, A., & Fern, O. (2006). *Intention recognition in the situation calculus and probability theory frameworks*. Paper presented at the Computational Logic in Multi-Agent Systems (CLIMA) Conference.

Freksa, C. (1992). Using orientation information for Qualitative spatial reasoning. In A. U. Frank, I. Campari & U. Formentini (Eds.), *Theories and methods of spatio-temporal reasoning in geographic space* (pp. 162-178). Heidelberg: Springer.

Harmelen, F., & McGuiness, D. (2004, 2004/02/10/). OWL Web Ontology Language Overview, W3C web site: http://www.w3.org/TR/2004/REC-owl-features-20040210/

Hoogs, A., & Perera, A. G. A. (2008). *Video Activity Recognition in the Real World*. Paper presented at the American Association of Artificial Intelligence (AAAI) Conference.

Jarvis, P. A., Lunt, T. F., & Myers, K. L. (2005). Identifying terrorist activity with AI plan-recognition technology. *AI Magazine, 26*(3), 9.

Jeon, H., Kim, T., & Choi, J. (2008). *Ontology-based User Intention Recognition for Proactive Planning of Intelligent Robot Behavior*. Paper presented at the International Conference on Multimedia and Ubiquitous Engineering Busan, Korea.

Kelley, R., Tavakkoli, A., King, C., Nicolescu, M., Nicolescu, M., & Bebis, G. (2008). *Understanding Human Intentions Via Hidden Markov Models in Autonomous Mobile Robots*. Paper presented at the 3rd ACM/IEEE International Conference on Human Robot Interaction Amsterdam.

Ligozat, G. (1993). Qualitative triangulation for spatial reasoning. In I. Campari & A. U. Frank (Eds.), *CPSIT 1993* (Vol. 716, pp. 54-68). Heidelberg: Springer.

Mao, W., & Gratch, J. (2004). *A Utility-Based Approach to Intention Recognition*. Paper presented at the AAMAS Workshop on Agent Tracking: Modeling Other Agents from Observations New York.

Martin, D., Burstein, M., Hobbs, J., Lassila, O., McDermott, D., McIlrath, S., . . . Sycara, K. (2004). OWL-S: Semantic Markup of Web Services, from http://www.w3.org/Submission/OWL-S/

Marvel, J., Hong, T.-H., & Messina, E. (2012). *2011 Solutions in Perception Challenge Performance Metrics and Results*. Paper presented at the Performance Metrics for Intelligent Systems (PerMIS) Conference, College Park, Maryland.

Moratz, R., Dylla, F., & Frommberger, J. (2005). *A relative orientation algebra with adjustable granularity*. Paper presented at the Proceedings of the Workshop on Agents in Real-Time and Dynamic Environments, IJCAI Edinburgh, Scotland.

Mulder, F., & Voorbraak, F. (2003). A formal description of tactical plan recognition. *Information Fusion, 4*(1).

Nau, D., Ghallab, M., & Traverso, P. (2004). *Automated Planning: Theory and Practice*. San Francisco, CA: Morgan Kaufmann Publishers Inc.

Newman, M., & Balakirsky, S. (2011). Contests in China Put Next-Generation Robot Technology to the Test. *IEEE Robotics and Automation Magazine, DOI 10.1109/MRA.2011.942540*.

Pereira, L. M., & Ahn, H. T. (2009). *Elder care via intention recognition and evolution prospection*. Paper presented at the 18th International Conference on Applications of Declarative Programming and Knowledge Management (INAP'09), Evora, Portugal.

Philipose, M., Fishkin, K., Perkowitz, M., Patterson, D., Hahnel, D., Fox, D., & Kautz, H. (2005). Inferring ADLs from interactions with objects. *IEEE Pervasive Computing*.

Randell, D., Cui, Z., & A., C. (1992). *A spatial logic based on regions and connection*. Paper presented at the 3rd International Conference on Representation and Reasoning, San Mateo, CA.

Ravi, N., Dandekar, N., Mysore, P., & Littman, M. (2005). *Activity Recognition from Accelerometer Data*. Paper presented at the Seventeenth Conference on Innovative Applications of Artificial Intelligence (IAAI/AAAI).

Roy, P., Bouchard, B., Bouzouane, A., & Giroux, S. (2007). *a hybrid plan recognition model for Alzheimer's patients: interleaved-erroneous dilemma*. Paper presented at the IEEE/WIC/ACM International Conference on Intelligent Agent Technology.

Sadri, F. (2011). Logic-Based Approaches to Intention Recognition. In N.-Y. Chong & F. Mastrogiovanni (Eds.), *Handbook of Research on Ambient Intelligence and Smart Environments: Trends and Perspectives* (pp. 346-375).

Schlenoff, C. (2012a). *An Approach to Ontology-Based Intention Recognition Using State Representations*. Paper presented at the Fourth International Conference on Knowledge Engineering and Ontology Development, Barcelona, Spain.

Schlenoff, C. (2012b). *An IEEE Standard Ontology for Robotics and Automation*. Paper presented at the International Conference on Intelligent Robots and Systems (IROS), Vilamoura, Algarve (Portugal).

Schlieder, C. (1995). Reasoning about ordering. In W. Kuhn & A. U. Frank (Eds.), *COSIT* (Vol. 988, pp. 341-349). Heidelberg: Springer.

Schrempf, O., & Hanebeck, U. (2005). *A Generic Model for Estimating User-Intentions in Human-Robot Cooperation*. Paper presented at the 2nd International Conference on Informatics in Control, Automation, and Robotics ICINCO 05 Barcelona.

Sukthanker, G., & Sycara, K. (2001). *Team-aware robotic demining agents for military simulation*. Paper presented at the Innovative Applications of Artificial Intelligence (IAAI).

Szabo, S., Norcross, R., & Shackleford, W. (2011). Safety of Human-Robot Collaboration Systems Project, from http://www.nist.gov/el/isd/ps/safhumrobcollsys.cfm

Tomasello, M., Carpenter, M., Call, K., Behne, T., & Moll, H. (2005). Understanding and sharing intentions: the origins of cultural cognition. *Behavorial Brain Science, 28*, 675-735.

W3C_Member_Submission. (2004). SWRL: A Semantic Web Rule Language Combining OWL and RuleML, from http://www.w3.org/Submission/SWRL/

Wallgrun, J. O., Frommberger, L., Wolter, D., Dylla, F., & Freksa, C. (2006). Qualitative spatial representation and reasoning in the SparQ-Toolbox. In T. Barkowsky, M. Knauff, G. Ligozat & D. R. Montello (Eds.), *Spatial Cognition V* (pp. 39-58): Springer.

Wolter, F., & Zakharyaschev, M. (2000). *Spatio-temporal representation and reasoning based on RCC-8*. Paper presented at the 7th Conference on Principles of Knowledge Representation and Reasoning (KR2000), Breckenridge, CO.

Youn, S.-J., & Oh, K.-W. (2007). Intention Recognition using a Graph Representation. *World Academy of Science, Engineering and Technology, 25*.

Keywords: ontologies, intention recognition, state representation, probabilistic, Region Connect Calculus 8 (RCC8), human-robot collaboration