Teaching a robot spatial expressions

Simon Dobnik and Stephen Pulman
Centre for Linguistics and Philology
Oxford University
Walton Street, Oxford, OX1 2HG, UK
{simon.dobnik|stephen.pulman}
@clg.ox.ac.uk

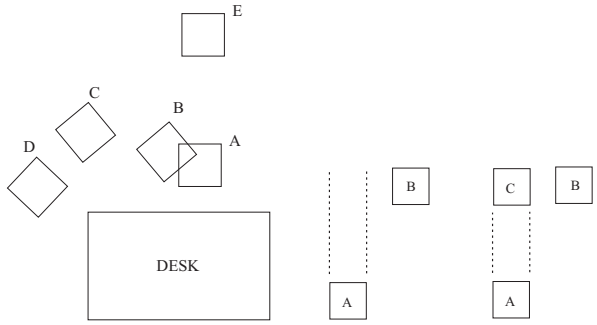
Paul Newman and Alastair Harrison
Department of Engineering Science
Oxford University
Parks Road, Oxford, OX1 3PJ, UK
{pnewman|arh}
@robots.ox.ac.uk

Abstract

Describing the semantics of natural language spatial expressions such as (1) 'moving forward slowly' and (2) 'you're in front of the desk' is not a straightforward task. In this paper we describe a setting with a mobile robot where the meanings of such expressions are learned from a set of robot data and natural language descriptions made by a human commentator. We start with simple robot-centered spatial expressions like (1). These do not make reference to the environment external to the robot. We then extend the learning to prepositional expressions like (2) which denote relations between the objects in the environment.

1 Introduction

A significant body of research in computational linguistics and artificial intelligence concentrates on interfacing information about the physical world and natural language expressions. Physical sciences have developed many methods for representing objects in space: points, vectors, representations of orientation and motion of objects. These representations are continuous and enable quantitative location of objects in a coordinate space. Any (coordinate) values will provide a precise and unique location of the objects. Natural language operates in a quite different manner. Spatial expressions partition the space into loose regions such as *near*, *back* and *left*. The location of the objects as well as the partitioning of the space is specified with a large degree of ambiguity. Which chair in Figure 1a is *in front of* the desk? How *near* is *near*? The choice of a natural language description also depends on other objects in the 'scene' ('distractors') and the perspective at which the scene is viewed. Is *B* behind *A* in both configurations in Figure 1b?



(a) (b)

Figure 1: *Ambiguity and distractors*

Factors like properties of objects (animacy, moveability) or focus affect the naturalness of spatial descriptions. Thus one would say that *Jane is near the house* but would probably not say that *The house is near Jane*. (Herskovits, 1986) lists many factors that may be important for the interpretation of spatial expressions.

Describing the semantics of spatial expressions is thus not a straightforward task, especially if done by hand. A good overall survey of various approaches is given in (Mukerjee, 1998).

We decided that a good test-bed for theories about the meanings of spatial expressions, particularly prepositions, would be a mobile robot setting. If a robot can be taught to 'understand' and use such expressions in a manner that would seem natural to a human observer, then we can be reasonably sure that we have captured at least something important about their semantics. In the robots we work with [Newman (2001 2002)], we have access to some aspects of the internal state of the robot, including its representation of speed and relative orientation, and its own representation of some salient features of the environment. Of course, as was stated above, we already know that this is not sufficient to be able to model the correct usage of natural language prepositions, but it gives us a platform on which we can superimpose successively more abstract properties in order to approach this more nearly. Furthermore, if the experiment is successful, we would have something of great practical utility: a robot that can both be instructed to move via natural language, and which can report what it is doing when moving under the control of another, or (more likely) moving autonomously (in a burning building or other hazardous environment, for example).

Our project is planned in several stages. At each stage, we collect a 'training corpus' of observations generated by the robot based on its internal state and representation of the environment, paired with a human generated commentary. The commentary consists of short phrases describing what the robot is doing. In the first phase, the internal state recorded is completely 'robot-centred', i.e. no reference to external properties of the environment is made. The commentary accordingly consists of phrases describing (from the robot's point of view) what is happening: e.g. slowing down; turning right; stopping. This commentary is obtained via a speech recognition system: a human narrator describes what is happening as the robot is driven around its environment.

In the second phase, the internal state is augmented with positional information about several objects in the environment, including the narrator. The commentary now makes reference to these objects and includes relative position and relational information: you are behind the desk; the desk is in front of me; etc. These situations carefully avoid the presence of 'distractors' so that the training data consists of clear instances of these spatial concepts.

In the third phase, distractors are included in the scenario. The commentary will sometimes mention these, and sometimes not. The aim is to see if the robot-generated narrative describing its route and relative position is able to select the most appropriate and informative set of prepositional relations in the presence of distractors.

In this paper we describe results obtained from the first two phases.

2 A description of the system

2.1 System design

MOOS [Newman (2001 2002)] stands for Mission Oriented Operating Suite and in its core represents a set of libraries and executables that run a mobile robot. Its main advantages are that it is platform independent (Unix, Windows) and that it is conceived as a modular system with a star-like topology. Each application within a MOOS 'community' has a connection to a single MOOS database (`MOOSDB`): it can publish as well as retrieve the information to the database but cannot publish or retrieve information directly from another client. There are two advantages of this design. Each application is independent of the others and can be written in a programming language of the author's choice. Natural language applications that are written in domain specific languages such as Prolog can be added to the MOOS community at no extra cost while preserving the benefits that these languages offer in terms of programme design. The communication is done through the TCP/IP protocol which means that the applications can run on a different machine than the `MOOSDB` so that the MOOS community can be spread over several different machines.

2.2 Experiment design

In the laboratory, a closed environment was set up with the following real-size objects: a chest, a box, a table, a pillar, a stack of tyres, a chair and a desk. The robot was left to explore the environment and build its map using a technique known as SLAM [Newman (2001)] which stands for simultaneous localisation and map building. Initially, the robot has no information about the environment or its location,

the SLAM application (pSMSLAM) builds its map incrementally through the observations of the environment. The robot can then use the map to navigate in the environment since it knows its absolute location. Figure 2 shows a photograph of the robot in the environment, Figure 3 contains a 3D graphical representation of the map built by the robot. Note that the robot is at a different location in each figure.

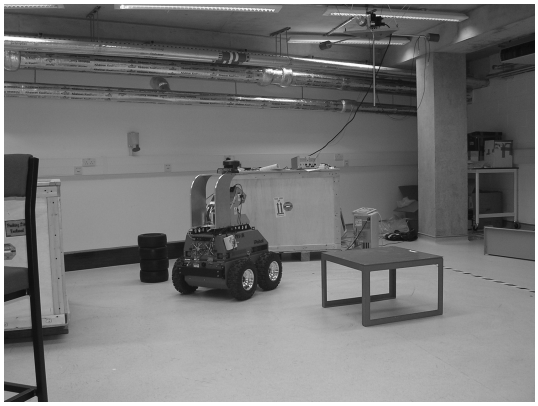


Figure 2: *The environment: the tyres are to the left of the robot, the table is to its right and the chest is behind it.*

The robot has no knowledge of discrete and discontinuous objects: the objects that can be recognised in Figure 3 by humans appear to it as sets of points. Objects were thus grounded manually. Each object was defined as a vector $\langle x, y, z, \text{object-name} \rangle$, where x, y and z are the coordinates of the central axis of each object relative to the robot's origin and *object-name* is the name of the object. A special MOOS application iCommentary was designed which recalculates the global object coordinates to relative object coordinates in respect to the current position and orientation of the vehicle as it is changing locations. The values generated by iCommentary are published to the MOOSDB at some predefined time interval.

During each experiment person A guided the robot performing motion in the environment using iRemote. iRemote accepts commands from a computer keyboard and publishes the values of desired thrust and desired rudder to the MOOS database. Person B commented the motion of the robot (Experiment 1) or the relations between the objects and the robot (Experiment 2) to iVoice, a speech recognition MOOS application based on Microsoft SAPI 5.1. The relational descriptions were grounded in the robot's perspective and their vocabulary was restricted to that listed in Section 3.1. A typical description would be 'The chair is behind you'. The recognised string was sent to the MOOSDB. iAGV provided the odometry information for the robot (Section 3.2), and pLogger produced a time-stamped log of all of the above values.

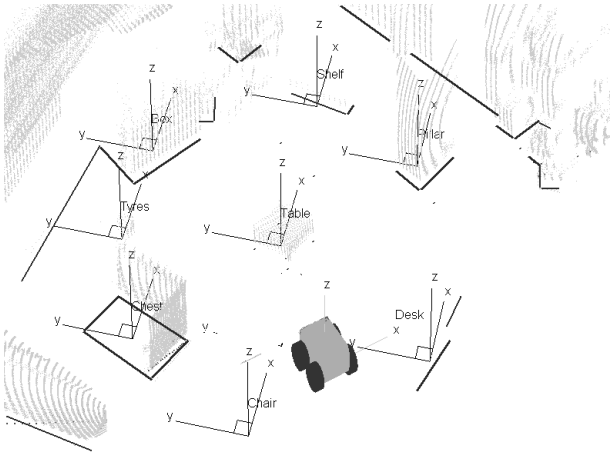


Figure 3: *A screen shot from the 'SLAM' process. The robot is equipped with a scanning laser which senses the local environmental geometry. Under an online, probabilistic framework, consecutive 'scans' are fused sequentially to produce a learnt 2D map (dark lines). The hood-like structure on the top of the robot in Figure 2 is a 3D scanning laser which produces clouds of 3D points shown in light gray. Only the most recently acquired 3D points are rendered — hence only two faces of the chest are discernible even though its entire circumference has been mapped. The labelled coordinate frames of objects used by the* iCommentary *process are also shown. As the vehicle moved, this process outputs the Euclidean relationships between the vehicle and all other labelled frames.*

3 The data

3.1 Language data

In Experiment 1 we restricted ourselves to a small set of spatial expressions that presumes no reference to the external environment. These are of three types: those describing the motion (*forward,*

backwards, *stopped*), those describing the manner of motion (*slowly*, *moderately* and *fast*), and those describing the direction of motion (*left*, *right* and *straight (on))*. All them were used 'dynamically' which means they were progressive descriptions of the routes that the robot was taking.

In Experiment 2 we included the reference to the external environment. *Relations* between the objects in the environment, the located object (LO) and the reference object (REFO), are expressed by natural language prepositions. These descriptions are static as they involve a fixed scene and include expressions such as 'A is *near* B', 'A is *behind* B', 'A is *in front of* B', 'A is *to the left/right of* B', 'A is *above* B', and 'A is *at* B'. The robot ('you') may or may not be one of the related objects A and B.

The expressions from Experiment 1 also allow for prepositional phrase modifiers that describe the location of movement. In this case the LO is always the moving entity (the robot or 'you') and the REFO is either the destination point of movement, as in 'Going forward/back/to the door', or its location, as in 'Turning left/right/at the table' and 'Stopping (near the window)'. In terms of reference to physical space, this class of expressions combines both kinds of expressions discussed above. Experiments to include these expressions are planned for the future stage of the project.

3.2 Non-language data

The non-language data includes information about the position and motion of the robot and the position of the objects. The information about the position and motion of the robot can be extracted from the odometry information published by `iAGV` to a log file. It includes the global x and y coordinates from some origin where the vehicle started, its heading in radians relative to the same coordinate frame, velocity in the x and y directions measured in m/s, and the angular velocity measured in rad/s. The speed of the vehicle can be calculated from the x and y coordinates by Pythagoras which means that it will always have positive values.

As described previously, we grounded the objects manually relative to the global frame, and then used the application called `iCommentary` to recalculate the coordinates to be relative to the robot's current location and orientation in regular time intervals. The values for the x, y and z coordinates of the robot in the data from `iCommentary` were thus $\langle 0, 0, 0\rangle$ at any given time, while objects such as the desk, for example, were given values such as $\langle 1.535, -2.558, 0.6355\rangle$ valid at some given time.

4 Machine learning

For machine learning we used Weka (Witten and Frank, 2000), a deservedly popular implementation of many common machine learning algorithms and associated tools written in the Java programming language. One big advantage is that it uses a unified file structure for the input data for all learning algorithms in the package. Different algorithms can be tested on one set of data without a difficulty. There is no need to pre-process the input to the learner each time and the results of various algorithm implementations can be compared on an equal basis.

4.1 The input data

Weka algorithms can handle datasets that consist of independent, unordered instances and where there is no relationship between the instances themselves. The software can read such datasets in the form of an ARFF file which is a structured text file. Each ARFF file contains a declaration of attributes (or classes of data). To learn a theory means to find relationships between the attributes that would correctly classify this (and new) data. One of the attributes must be chosen as the target concept or the class that the learned hypothesis should predict. Attributes can be numeric or nominal, in the later case the set of possible nominal values must also be declared. Instances are represented as vectors of attribute values $\langle$Attribute1, Attribute2, Attribute3$\rangle$, the last attribute value being by convention the target concept that we want to learn.

The choice and representation of attributes has a highly significant impact on learning. In a MOOS log file, each attribute is logged separately, one per line, each entry with its own time stamp. The first pre-processing step for data from both experiments was to find non-linguistic spatial data that corresponded to the time when the description was logged. This means that the time of each description defines an instance, whereas the linguistic description of robot's movement (Experiment 1) or the

preposition used to describe the relation between LO and REFO (Experiment 2) are the target concepts that are learned. The learned theory is a set of hypotheses that relate non-linguistic data to linguistic descriptions at any time.

What are the the attributes from each experiment? In Experiment 1 each instance was represented as a vector of attribute values ⟨Time, DeltaHeading, Heading, Motion, Manner, OriginalDescription⟩. The DeltaHeading attribute (angular velocity in Section 3.2) refers to the rate of change in heading measured in rad/s. MOOS headings are zero for north, positive for left and negative for right. This is because this attribute is a measurement of the robot's yaw axis, rather than conventional compass headings. The Speed attribute refers to the speed of the vehicle in m/s. As noted in Section 3.2 the measure can only have positive values. It is desirable to have positive values to indicate forward movement, and negative for reversing. Additional calculations were required to determine the +/- prefix: we omit the details here.

The following three attributes Heading, Motion and Manner refer to the three groups of spatial expressions that we identified in Section 3.1. Because these are nominal attributes they will have a closed set of values: Heading : {straight, right, left}, Motion : {stopped, forward, backward} and Manner : {none, slowly, moderately, fast}. The values were obtained by parsing the fourth attribute OriginalDescription which is a string published by *iVoice* after recognising a voice utterance. The utterance did not necessarily include all the attributes. In this case, a default value was assigned to that attribute. For example, for the attribute Manner, the value 'none' was assigned if the utterance did not include a specification of the manner of motion; if no word for the attribute Motion was specified (as in *You're moving right now*), the value was defaulted to 'stopped', and if no input for the attribute Heading was found (as in *You're moving forward*), the value was defaulted to 'straight'. This introduced some noise into the training data.

Experiment 2 included learning a relation between LO and REFO. The location of LO and REFO in relation to the vehicle (which may be one of these objects), their x and y coordinates, can be extracted from the data from `iCommentary`. Preposition : {behind, to-the-right-of, to-the-left-of, in-front-of} is the target concept to be learned. Thus, in this case each instances is represented as a vector of the following attribute values ⟨Time, LO-x, LO-y, REFO-x, REFO-y, Preposition, OriginalDescription⟩.

In both experiments the attributes Time and OriginalDescription were not included in the learning scheme; they were kept in an ARFF file for debugging purposes to be able to relate the information between this file and the log file.

4.2 The learning algorithms used

For the learning exercise, two frequently used algorithms in the Weka toolkit were chosen: J48 and Naive Bayes. They use quite different approaches in forming hypotheses about the data.

J48 is Weka's implementation of the C.45 decision tree learner.[1] J48 uses a method that (Witten and Frank, 2000) describe as 'divide and conquer'. First, an attribute is selected as a root node for the tree and then branches are created for each of its possible values. The process is repeated recursively for each branch, using only those instances that can be classified under that branch. If all instances at a node have the same classification, the development of that part of the tree can be stopped. The difficulty is to decide at which attribute to split on. J48 uses a measure called *information gain* measured to accomplish this. For details, see (Witten and Frank, 2000), p.89ff.

Naive Bayes on the other hand learns probabilistic knowledge. Instead of choosing just one attribute to split on, it uses all attributes and allows them to make contributions to the decision as if they were all equally important and independent of one another. Although in real datasets attributes are not equally important and independent of one another, Naive Bayes nonetheless works surprisingly well in practice. One disadvantage of the approach is that the probabilistic knowledge is not very informative to the plain human eye. Decision trees on the other hand are much more readable to humans.

[1] It implements the last public version (Revision 8) of the C4.5 algorithm, before C5.0 commercial implementation was released.

5 Results

5.1 Experiment 1: descriptions of motion

There were 82 instances in the training set. Weka was set to use 10-fold cross-validation to test the learned hypotheses which means that 90% of the instances (74) were used for training on each pass.

Both algorithms performed very well on learning the Heading attribute. The accuracy of classification after 10-fold cross-validation was 97.6% for J48 and 98.8% for Naive Bayes. J48 produced a decision tree in Figure 4. The numbers in brackets on the leaf nodes indicate the number of instances that have been classified under that node. A lesser accuracy was obtained for the other two attributes as shown in Table 1.

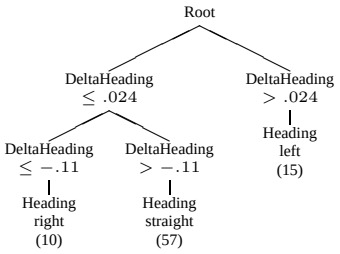


Figure 4: *A decision tree for the Heading attribute*

Learner	Motion	Manner
J48	82.9%	70.7%
Naive Bayes	72%	67%

Table 1: *Correctly classified instances*

Figure 5 shows a simplified decision tree for the Manner attribute. The learner chose to split the data on the Motion attribute first. This is expected since negative Speed values mean backward motion. The classification of the Manner attribute is thus different if the Speed attribute is negative or positive: fast backward motion has the lowest rather than the highest value. Under the Motion = 'forward' branch the split on the Speed attribute appears reasonable: the leaves under the Speed $\leq$ .3 branch (simplified here to exclude a further differentiation on the DeltaHeading attribute) include 'none', 'slowly' and 'moderately', whereas the terminals under the Speed $>$ .3 branch include 'moderately' and 'fast'. Note also that the second number in the brackets on the leaves tells us the number of incorrectly classified instances under that node.

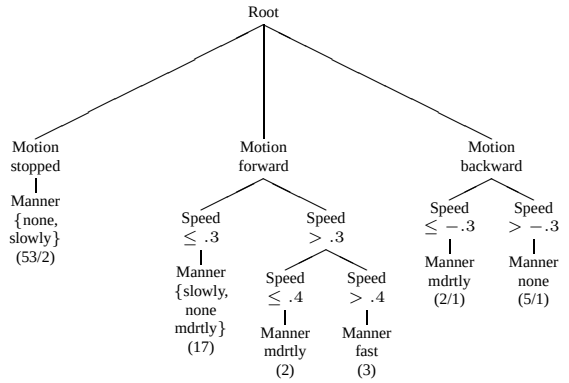


Figure 5: *A decision tree for the Manner attribute*

5.2 Experiment 2: Prepositions

The training set in this experiment contained 251 instances. Again 10-fold cross-validation was used to test the learned hypotheses. The performance of the trained model is comparable to the models learned in Experiment 1: the learned decision tree classifies 74.9% instances correctly whereas the Naive Bayes model does so for 77.3% of instances.

Figure 6 shows a decision tree that was learned. Because this learning task involves more attributes than the previous one, and because an assumption is made that the attributes are independent of each other (and hence the learner does not take LO-x and LO-y to be related), it is more difficult to observe the motivations behind the learner's choice. For example, according to our ideal notions of the meaning of the prepositions in question, there should be no difference in classifications if LO-x $\leq .8$ or LO-x $> .8$. The classification tells us about the structure of our training set. The preposition 'to the right' occurs only if LO-x $> .8$, whereas the preposition 'in front of' occurs under both branches. Perhaps, the commentator used the preposition 'to the right' mostly in a situation where the REFO was the robot (REFO-x ≤ 1.6) and the LO was an object to its right (LO-x $> .8$ and LO-y ≤ 1.9). From this, can we speculate that if one wants to ground the robot (LO), one is more likely to choose objects to its left rather than objects to its right? Another example that shows the influence of the training set data is the branch 'lo-x > -1.4'. Under this branch, one does not

have to look at the coordinates of the REFO in order to choose a preposition. If LO-x ≤ -1 and LO-y > -1.4 (roughly corresponding to the top left quadrant of the coordinate system) then the most likely preposition is 'to the left', regardless of the coordinates of the REFO. Finally, it is important to note that the numerical values that have been learned are applicable to the scale of the scene in which the descriptions were made: the typical distances between the robot and other objects in the scene. With a change of scene, these should be scaled appropriately. We have not yet implemented this requirement.

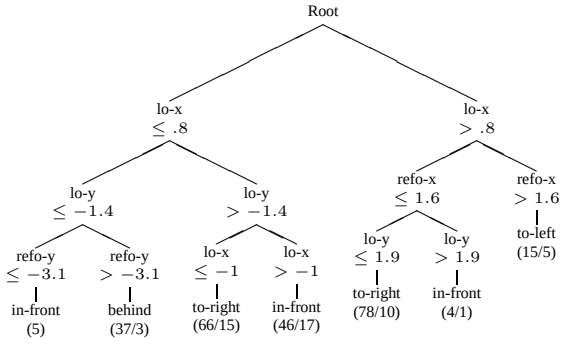


Figure 6: *A decision tree for prepositions*

In conclusion, machine learning (as it is set up in these experiments) cannot induce the semantics of prepositions that would be informative to humans, but can 'teach' the robot to have a sufficient notion what these prepositions in a given environment mean.

6 Robotics, machine learning and descriptions of space

In this section we briefly discuss two approaches that combine machine learning with description of space. (Lauria et al., 2002) describe a setting with a mobile robot that uses a technique known as Instruction-Based Learning (IBL) to learn route descriptions in a miniature town. A robot has a preprogrammed set of primitive descriptions such as 'move forward until [(past|over|across) ⟨landmark⟩]' which it can execute. IBL is a special kind of machine learning where the robot engages in a dialogue with the user, and then, depending on whether it knows how to perform the task, either performs it, or asks the user to explain how to perform it by using the expressions referring to the previously known tasks. The knowledge of the new new task is acquired through instruction and the new task is subsequently added to its database. IBL is a different kind of machine learning in comparison to the algorithms used in this paper. What we are trying to do here is to ground the basic concepts used by the robot in properties of the internal state of the robot itself. The Weka algorithms we use try to find a set of attribute-values (properties of the robot and the environment) that predict another attribute (a natural language description). IBL, in contrast, is compiling high level instructions to lower level instructions whose grounding is taken for granted.

The learning tasks in (Roy, 2002) are similar to the ones described in this paper. The domain of the target concepts that are learned is wider than ours. Each training instance consists of a natural language sequence of words and a set of valued features that correspond to the semantics of this bag of words. The system has no prior knowledge of word's lexical semantics, word classes nor the syntactic structures that they occur in. All are automatically acquired by probabilistic learning algorithms from a set of features and words. The experiments are based on a rectangle description problem where images of coloured rectangles are automatically generated and human subjects are asked to describe the scene. Our approach differs from (Roy, 2002) in that we transpose the learning task to the domain of a mobile robot, we use a different set of spatial features and different learning algorithms. Nevertheless, many of our aims are similar and we intend to follow this work closely.

7 Improvements and further research

The 70–80% accuracy could be improved to a higher rate, perhaps to that achieved for the attribute Heading in Experiment 1, by increasing the size of the training set. By comparison with most machine learning experiments our training sets are very small.

The expressions have been learned with an acceptable rate of success. The learned knowledge should be tested in a 'conversation' with the robot, for example, to instruct the robot to perform motion in the environment or to ground objects. In both

cases, however, there seems to be a great deal of ambiguity that need to be resolved before any action is performed. For example, if a robot is instructed to turn left, there is a range of attribute values that correspond to 'left', but only a particular one has to be chosen to perform the action. Equally, if the robot is to be able to ground (or name) the object on its left by being told 'The table is to the left of you', the learned models would return a set of candidate objects from which a single object must be chosen.

This problem is related to the problem of 'distractors' for prepositional expressions. In the present experiments distractors were ignored and a commentator attempted to describe the location of an object by relating it to any other object in the scene. However, humans do not ground objects this way: in order to locate an object they choose a REFO that is salient or has properties such as being movable or animate, depending on the relation/preposition in question. Representing objects as vectors of values of their x, y and z coordinates is thus too simplistic. Additional features can be added to our system incrementally and machine learning should tell us more about their significance for individual prepositions.

With distractors included, the learning experiment can be formulated in three different ways: (1) given LO and REFO, which preposition relates them best; or (2) given REFO and a relation such as 'left', ground the LO; or (3) given LO and a relation, find a REFO. The formulation of the learning task depends on how the learned knowledge is going to be used. For example, (1) would be used when the robot attempts to describe the environment, (2) for grounding unknown objects and (3) for describing a scene whereby the relation is specified. The current non-symbolic machine learning techniques do not allow us to combine the knowledge acquired in all three settings in a single model. This would be possible with a symbolic learning approach such as Inductive Logic Programming (ILP) (Muggleton and De Raedt, 1994) which we are planning to investigate in the future.

8 Conclusion

In our joint work between computational linguistics and robot engineering so far a significant amount of time and effort was spent on understanding each other's language and on interfacing our software and systems. In this paper we report of the stage where we are able to learn simple descriptions of movement and spatial relations between the objects in the environment using non-symbolic methods. We are planning to increase the number of instances in our training data to improve the accuracy of learning, show that the learned knowledge can be used by a robot to describe its environment, and include additional properties of the environment to be able to take into account the presence of 'distractor' objects.

References

M. W. M. G Dissanayake, P. M. Newman, H. F. Durrant-Whyte, S. Clark, and M. Csorba. 2001. A solution to the simultaneous localization and map building (SLAM) problem. *IEEE Transactions on Robotic and Automation*, 17(3):229–241.

Annette Herskovits. 1986. *Language and spatial cognition: an inderdisciplinary study of the prepositions in English*. Studies in natural language processing. Cambridge: Cambridge University Press.

S. Lauria, T. Kyriacou, G. Bugman, J. Bos, and E. Klein. 2002. Converting natural language route instructions into robot-executable procedures. In *Proceedings of the 2002 IEEE International Workshop on Robot and Human Interactive Communication (Roman'02)*, pages 223–228, Berlin, Germany.

Stephen Muggleton and L. De Raedt. 1994. Inductive logic programming. theory and methods. *Journal of Logic Programming*, 19(20):629–679.

Amitabha Mukerjee. 1998. Neat versus scruffy: a review of computational models for spatial expressions. In Patrick Olivier and Klaus-Peter Gapp, editors, *Representation and processing of spatial expressions*. Lawrence Erlbaum Associates, Inc.

Paul Newman, 2001-2002. *MOOS - Mission Orientated Operating Suite*. Oxford University Robotics Research Group, Department of Engineering Science, Oxford University, http://www.robots.ox.ac.uk/~pnewman/MOOS/.

Deb K. Roy. 2002. Learning visually-grounded words and syntax for a scene description task. *Computer speech and language*, 16(3):353–385.

Ian H. Witten and Eibe Frank. 2000. *Data Mining: Practical machine learning tools with Java implementations*. Morgan Kaufmann, San Francisco.