Object Selection using a Spatial Language for Flexible Assembly*

Batu Akan, Baran Çürüklü, Giacomo Spampinato, Lars Asplund
Mälardalens högskola, Box 883, 721 23 Västeras, Sweden
{batu.akan, baran.curuklu, giacomo.spampinato, lars.asplund}@mdh.se

Abstract

In this paper we present a new simplified natural language that makes use of spatial relations between the objects in scene to navigate an industrial robot for simple pick and place applications. Developing easy to use, intuitive interfaces is crucial to introduce robotic automation to many small medium sized enterprises (SMEs). Due to their continuously changing product lines, reprogramming costs are far more higher than installation costs. In order to hide the complexities of robot programming we propose a natural language where the use can control and jog the robot based on reference objects in the scene. We used Gaussian kernels to represent spatial regions, such as left *or* above. *Finally we present some dialogues between the user and robot to demonstrate the usefulness of the proposed system.*

1 Introduction

Companies producing mass market products such as car industries have been using industrial robots for machine tending, joining, and welding metal sheets for several decades. Thus, in many cases an investment in industrial robots is seen as a vital action that will strengthen a company's position in the market since such investments will increase their productivity. However, in small medium enterprises (SMEs) robots are not commonly found. Even though the hardware cost of industrial robots has decreased, the integration and programming costs make them unfavorable for many SMEs. In order to make industrial robots more favorable in the SME sector, the issues of flexibility has to be resolved. Typically for those SMEs, that have low volume production and frequently changing applications, it is quite expensive to afford a professional programmer or technician, therefore a human robot interaction solution is strongly demanded in order to let the user to program these systems in an intuitive way. Using a high-level natural language, which hides the low-level programming from the user, will enable a task expert who has knowledge in manufacturing process to easily program the robot and let the robot to switch between previously learned tasks. Thus, the goal is to eventually bring robot programming to a stage where it is as easy as teaching the task to a new member of the work team.

One of the problems associated with this idea is that, how the structure of the language can be kept as simple as in daily spoken language, while maintaining the accuracy and precision required for industrial applications. Our daily spoken language is extremely powerful. There are several ways of saying even the simplest thing. A conversation in a natural language is strongly connected to individuals involved, the context and even the culture. All aspects obviously mean that we simply can not treat a full natural language as a mean of communication with an industrial robot system.

There is a large and diverse literature on spatial representation in humans and other species. Landau and Jackendoff [8] gives a detailed overview of the spatial terms and cognition. They divide spatial processing into object recognition, and specifying paths and locations. In this paper we focus on object recognition and specifying locations for selection of object in cluttered scenes.

Spatial relations have been widely used in mobile robot navigation. Skubic et al. [10] investigated the use of spatial relationships to establish a natural communication between people and robots. Using linguistic spatial terms, a high-level spatial description is generated which describes the overall environment, and a detailed description is also generated for each object. Their work mostly focuses of positions of the objects in the environment, relative to the robot. Tellex and Roy [12] uses a set of spatial routines to develop a speech controlled wheelchair that understands high level natural language commands. In their work the same command may lead to different actions depending on the environment. Gorniak et al. [9] propose a situated language to be used in computer games, where the semantic meaning of the commands depend on the situation.

Spatial commands is used to select objects in the scene as well. Sugiyama et al. [11] in their work, propose a system that indicates to a listener which object is currently under consideration by using pointing gestures and reference terms as 'this' and 'that'. In cases of ambiguity they use other properties of the objects, such as color, as a reference term. Kurnia et al. [7] makes use

*This work described in this paper is supported by *Robotdalen* and *Sparbanksstiftelsen*.

of spatial commands in order to help the robot to select or recognize certain objects in the scene. In their work they make use of color, relative size and relative shape as input features. The system asks the user questions in way minimize the least number of questions. Depending on the user's answer the system eliminates irrelevant objects and asks a new question to the user until the number of possible objects are reduced to one.

In the case of industrial robots Haage et al. [4] developed a prototype that can manipulate a robot through speech interface. In their work they provide commands for jogging the robot, and adding and removing control points to define and refine a path.

Spatial terms and languages have often found application area in mobile robotics, where the world can be represented in two dimensions (2D) and the movement of the robot is restricted in this 2D plane. However industrial robots can move in a three dimensional (3D) world, and yet capable of more complex movements compared to mobile robots. The purpose of this paper is to introduce spatial relations in to the world of industrial robot programming, with the hope of making the programming phase more easy and intuitive.

The organization of the paper is as follows; In Section 2 we give an overview explanation of proposed system architecture; automatic speech recognition, simulation environment, spatial terms and the reasoning system. In Section 3 we present test cases and Section 4 provides discussion and conclusion.

2 Architecture

The proposed system has an speech driven interface. The user commands via voice and receives feedback from the robot in speech format. The commands are parsed and passed to the reasoning system which eventually drives the robot to perform the necessary actions and commands. requested by the user. Figure 1 presents an overview of the system architecture.

2.1 Speech Recognition

In this project we used Microsoft Speech API 5.1 (SAPI 5.1) [6], both to recognize the commands given to the robot and to synthesize speech in order to give feedback to the user. Even though the automatic speech recognition (ASR) engines has come a long way, they are still far from perfect. SAPI 5.1 functions in two modes; Free dictation mode, and command & control (C&C) mode. In free dictation mode, the ASR engine recognizes word in any context, therefore recognition accuracy is low, because the engine is selecting from a huge vocabulary set, that is not constrained by any grammar rules. On the other hand command and control mode works through a set of grammar rules and a limited vocabulary therefore increasing the accuracy of recognition process. In this project C&C mode is used with a hand generated grammar rules, based on the natural language processing (NLP) grammar to listen to user commands. For providing feedback to the user, the text to speech (TTS) functions of SAPI is used.

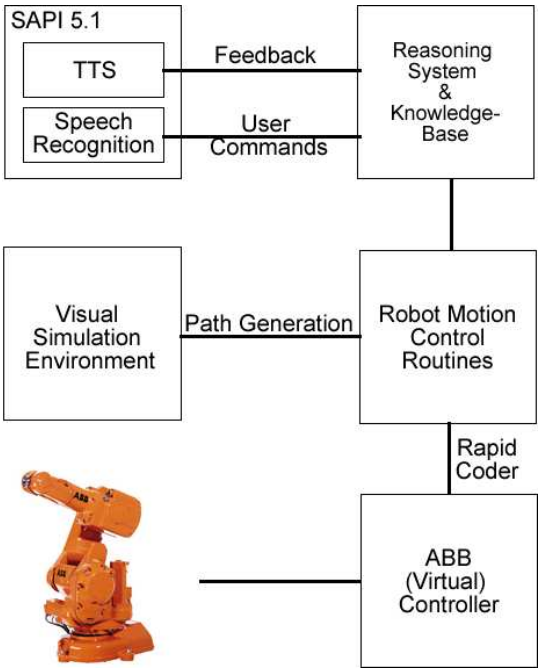

Figure 1. Block diagram of the proposed system.

2.2 Visual Simulation Environment and High Level Movement Functions

A simulation environment which can simulate a general 6 Degree of Freedom (6DoF) industrial robot has been created in an OpenGL environment in order to simulate the behavior of the robot in a noise free environment where objects' positions and orientations are known. Besides providing a quick test environment, it also acts as a visual feedback to the user. The simulator wraps around basic movement functions of ABB RAPID [1] language. Basically the simulator provides a similar programming interface as the RAPID language provides. Upon calling the robot manipulation functions including basic movement and gripper functions, these calls are first tested for reachability along the path. If a point along the path is not reachable, or the orientation requested is not achievable or if the robot exceeds its joint limitations in any part of the path then the test fails. If the reachability test fails then the command is rejected from the queue, otherwise it is added to the job queue and the path of the tool center point is shown in the OpenGL environment. When the "execute" function is called all the functions in waiting the queue are played in the simulator and if the application is connected to an ABB controller then the queued functions are converted to a RAPID module and send to the controller and executed automatically in the virtual or the real controller.

On a higher level of abstraction, commands issued by the user, such as picking up an object is realized

through a set of low-level movement functions. Given the position, the orientation of an object and the gripping pattern to be used in order to pickup that object, the Higher Level Movement Commands generates the necessary sequence of low-level movement functions for approaching, grasping and retracting. If any of the sub commands fail the test for reachability in the given grasping pattern, then the high level command and all the sub commands are revoked from the execution queue. A failure code is returned back to the reasoning system, where it can ask the user to select another object to pick or try to pick the object using a different grasping pattern.

2.3 Spatial Terms

In English, representation of an objects position/location requires three elements; the object to be located or namely *figure*, the reference object and their relationship [8]. The figure and the reference object are encoded as noun phrases and the spatial relationship between these objects are encoded as prepositional phrases that clarifies in which *region* the figure is in, in respect to the reference object.

In our daily spoken language prepositional terms often have a vague meaning. The meaning of spatial terms such as; *left* or *behind* cannot be divided into discrete regions, it is difficult to determine where the notion of *left* starts and where it ends. The size as well as other structural properties of the figure or the reference object affects the meaning encoded in the sentence. For example two pencils can be regarded as, one being on the left of the other one if the distance between them is relatively small, like e.g. 30-40cm. On the other hand two automobiles that are meters apart from each other can be regarded as next to each other as well. Also depending on the context such spatial notions can lose their meaning or helpfulness if the spatial relation between the reference and the figure are relatively weak or dominated by other spatial relations. For example in Figure 2, it can be easily seen that the green object is on the right of the red object. But it is not clear that one can call the blue object to be on the right of a red object even though it lies on the same vertical line as the green object. In this case *behind* region dominates over *right* region.

In order to overcome the complexities stated above we represent the spatial regions using Gaussian kernels, hence turning the problem into a multi-class classification problem. The object is assigned to a region represented by the respective Gaussian kernel based on its Mahalanobis distance to the kernel [3]. If the objects Mahalanobis distance is close to both kernels such as left and front then it is assigned to both. Figure 3 shows four Gaussian kernels overlaid around a reference object. According to the figure the green object is on the right of the red object and the blue object is in front of the red object. A multivariate Gaussian kernel can be expressed as,

$$N(\mu, \Sigma) = \frac{1}{(2\pi)^{N/2}|\Sigma|^{1/2}} e^{\left(-\frac{1}{2}(x-\mu)^\top \Sigma^{-1}(x-\mu)\right)}. \quad (1)$$

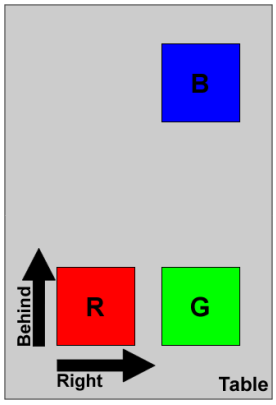

Figure 2. Example showing weak spatial relations.

The Gaussian kernel is governed by two parameters, the mean μ and covariance matrix Σ. One can change the size of the region governed by the respective Gaussian by adjusting the values. This is very advantageous for representing spatial terms. The mean and the covariance can be adjusted depending on the size of the reference object. If the reference object is the workbench or the robot itself then larger values for mean and covariance should be selected, if the reference object is work-objects or relatively small sized tools then smaller values should be used for the mean and the covariance. Also by adjusting the shape of the Gaussian kernel, it is possible to give more importance to some attributes of the spatial term. For example for a sentence like "Pick up the objects from the container on your left", an sphere like ellipsoid would give a better representation of the region, where as for a sentence like, "Put the objects along a line" a very elongated ellipsoid would be more advantageous.

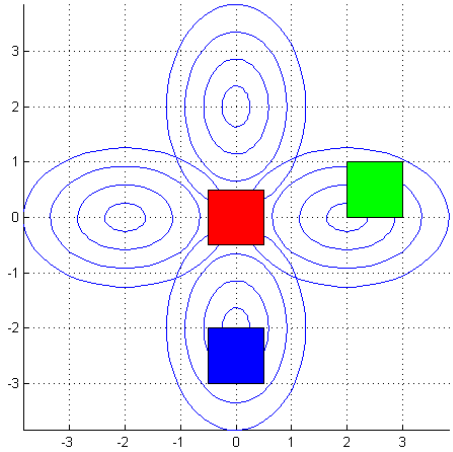

Figure 3. A view from the top of the table, overlaid with gaussian kernels representing left, right, behind and infront regions for the red object.

In this paper we implemented the spatial terms; behind,

in front of, to the left of, to the right of, over, and under. A full list of prepositions in English can be found in Landau's work [8].

2.4 Knowledge Base and Reasoning System

Reasoning system functions as the system brain. While interpreting the speech commands given by the user it plans and calls necessary motor functions to realize the commands. For this project a Prolog engine has been selected to implement the knowledge-base and the reasoning system [13]. Prolog is a declarative programming language with its roots in logic. A call to Prolog with a predicate becomes a statement that the Prolog engine tries to proof. Sometimes a call can results in more than one solution. This means that the Prolog engine can return more than one solution to a given query while it tries to prove the correctness of a statement. We found this unique property of the language as well as its simple grammar valuable enough to motivate its use as system brain in this project.

As objects are inserted into the simulation environment, or recognized by vision system which is currently under development, information about the objects are asserted into the knowledge-base. Only structural relations about the objects are kept in the knowledge-base, where as, position and orientation are not stored. Spatial relations between the objects are also not stored in the knowledge base. They are computed by the visual simulation environment on demand. This makes the knowledge-base more manageable otherwise the number of relations between the objects would grow exponentially as the number of objects increase. The most important predicate in the knowledge-base is the *property* predicate. It is used to define structural information of the objects. It can also be used to define the type of an object; such as defining whether the object is a workbench or a manufacturing object or of any other object type [2]. Based on these types different actions can be allowed on this object. A manufacturing-object can be flagged as safe for picking up, where as a workbench should not be picked up. An example definition for a red, cylindrical, manufacturing object is as follows:

```
object(object12). % object identifier
property(object12, color, red).
property(object12, shape, cylindrical).
property(object12, type, manufacturing).
```

Also sub-locations for these objects can be defined as a property. Defining sub-regions are also defined similarly to the objects. Their location and orientation relative to the parent object are kept in the simulation environment. A cylindrical hole that is a sublocation of object20 can be defined as;

```
object(object21).
property(object21, sublocation, object20).
property(object21, shape, cylindrical).
```

Spoken command	*Purpose*
pick up	Picks up a specified object
put selection/it	Puts the object
apply selection	Ends selection phase
it is/they are	Declares new properties for the selection
execute	Begins executing the path
cancel	Clears out the execution queue

Table 1. List of commands to control the robot.

The reasoning system receives text input from automatic speech recognition engine and converts it into Prolog predicates using the set of grammar rules defined by the Natural Language Processor (NLP). These generated predicates capture information about the structural properties of the object such as color, shape or any other property that is defined for that object type as well as cues revealing relative spatial positions of the objects in the scene. A command like: "Pick up a red object which is behind a cubic object.", would translate to Prolog predicates as:

```
object(X), property(X, color, red),
object(Y), property(Y,shape, cubic),
spatialpos(X,Y,behind), pickup(X).
```

Based on the predicates generated, the reasoning system searches for possible matches. As the user gives new descriptions, the engine updates its search criteria with this new information until only one object remains or the user approves the selection by saying "Apply to selection". In case of an ambiguity or an error the reasoning system initiates a call to the text to speech engine and asks the user to clarify the situation.

In the proposed language there are commands for object manipulation, object selection, and managing the job queue. Table 1 shows the list of available commands in command set. Manipulation and declaration phrases are followed by a noun phrase describing the structural properties of the object or they are followed by prepositional phrase, giving a reference object and its spatial relation with the *figure* object.

3 Experimental Results

In order to demonstrate the usefulness of the proposed system, we designed test cases that utilizes the industrial robot for simple palletizing/depalletizing tasks. The simulated testing environment consists of one ABB IRB-140 robot, one workbench and two conveyor bands on each side of the robot. Figure 4 presents a view of the robot and the working environment around it. In Figure 4 the robot is asked to pick up all blue objects and put them on the conveyor to the left of the robot. Over the workbench 18 manufacturing-objects with 4 different colors and 2 different shapes and a cassette with 4 holes of 2 different shapes are present. Figure 5 shows a detailed layout of the objects on the workbench.

The Gaussian kernel parameters for objects are entered manually depending on the size of the object, so that the kernels span a region in space twice the width of the object in the relevant direction.

During the experiments we asked the users to select and pickup a specific manufacturing object from the workbench and put it either in the cassette or over one of the conveyor-bands by trying to define the object using the spatial commands. Since the testing environment is a simulation and we have no information about where the user is the commands are given in reference to the robot itself, which means that the user should adjust the commands as if he/she is standing in the place of the robot.

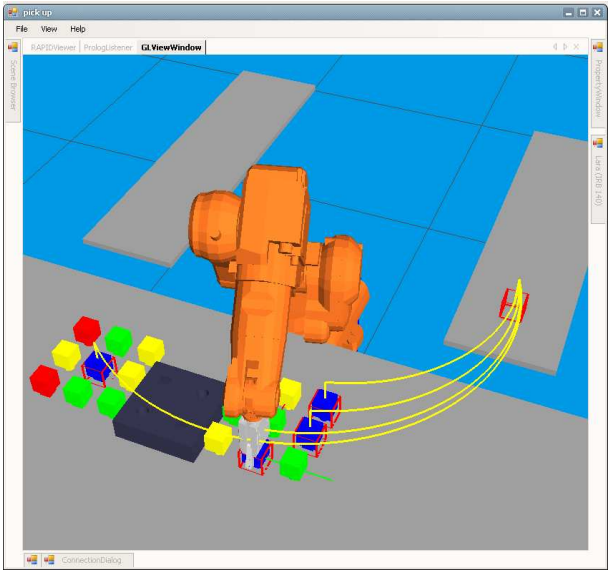

Figure 4. Screenshot of the simulator where the robot is asked to pick and place all the blue objects over the conveyor band.

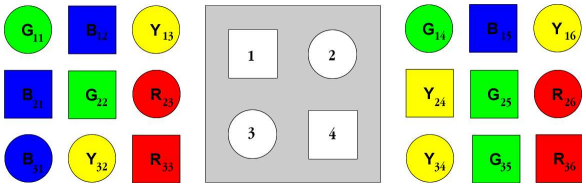

Figure 5. Layout of the objects

A dialog between the robot and a user is given below. In this given case the user is trying to pick up the green object (G_{11}) on the upper left corner. The user does not specify the color or the shape of the object but only relies on its spatial relation with the surrounding objects.

User: Pick up an object.
Robot: I see five objects that fit the description. Which one?
User: It is behind a blue object.
Robot: I see two objects that fit the description.
User: It is on the left of a blue object.
(One object remains in the selection therefore a path is generated and showed to the user in the simulation window)
User: Execute
Robot: *(picks up the desired object)*
User: Put it over the conveyor band
Robot: I see two objects that fit the description. Which one?
User: It is on the left of you.
User: Execute
Robot: *(Leaves the object over the conveyor band.)*

A second case to present is when the user wants to select the yellow object (Y_{16}) which has the exact same immediate and secondary neighbors as the yellow object (Y_{13}). It is, thus, more difficult to describe the object through its immediate neighbors, however its relation with the cassette in the middle can be used to select the object, since the cassettes Gaussian kernels are big enough to cover the area for the object (Y_{16}). The dialog is given below:

User: Pick up a yellow object.
Robot: I see five objects that fit the description. Which one?
User: It is behind a red cylindrical object.
Robot: I see two objects that fit the description.
User: It is on the left of a cassette.
User: Put it in hole three
User: Execute
Robot: *(picks up the desired object and puts it in hole three).*

On a higher level of abstraction the user may operate on a selection of objects. He/she may pick up all the green objects on the workbench and put them into a matching hole in the cassette. This kind of a dialog is more close to palletizing or assembly. The cassette may be seen as an object to be assembled or a palette where the objects are to be put into. The dialog between the user and the robot is given below;

User: Select all red objects.
Robot: I see four objects that fit the description.
User: Apply selection.
Robot: Four objects are selected.
User: Put selection into cassette.
User: Execute
Robot: *(picks up the selected objects and puts them into the cassette by respecting the shapes)*

4 Discussion

In this paper, we demonstrated a high-level restricted language in order to command an industrial robot for simple pick and place applications. The proposed language can handle attributes of the objects in the environment, such as shapes, colors, and other features in a natural way, it is also equipped with functions for handling spatial information enabling the user to be able to relate objects spatially to static objects, robots, table, etc. as well as to other work objects.

The proposed language may seem limited in its vocabulary and command set, however, as in the case with natural languages it is possible to describe complex patterns through use of the proposed command set. The system allows the user to expend the knowledge base as well. Thus, it is possible to add previously unknown properties of manufacturing objects as well as new manufacturing objects.

In the near future we intend to increase the scope of this language used for human robot interaction as well as introduce a gesture based system that relies on hand movements to overcome some of the problems introduced by using a speech driven system [5] . In the current model there are no distinctions between two objects inside a spatial region, e.g. if two objects fulfill the relationship "to the left" it is not straightforward to say which one is "more to left" or "closer to" the reference object. This problem can be solved by changing the size of the Gaussian kernels iteratively. Yet another restriction in the current language is that solely the objects are represented, neither the workbench or other types of objects, such as manufacturing machines and tools used in the manufacturing process are represented. Adding these objects into the model is essential. A typical command using the workbench would be "Move all red objects to the upper left corner of the workbench" or "Clear the center of the workbench". In most cases there will be a CNC machine in the robot cell, thus there is a natural need of increasing the vocabulary with words associated with such machines. Sensors are becoming an essential part of a industrial robots and a robot cells. We are in the of a process of integrating a 3D camera with the industrial robot. The 3D camera system will be integrated with the proposed simulation environment. Obviously there are a number of challenging research questions associated with this approach. The environment created in this way will consist of virtual as well as real objects and machines, tools. Handling these relationships will require a richer language. Another essential part of the work that is in front of us is to test the system on broad range of users.

References

[1] ABB Flexible Automation, 72168, Vasteras, SWEDEN. *RAPID Reference Manual 4.0*.

[2] B. Akan, B. Curuklu, and L. Asplund. Interacting with industrial robots through a multi-modal language and sensor systems. Seoul, Korea, October 2008. International Symposium on Robotics.

[3] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification*. Wiley Interscience Publication, 2nd edition, 2001.

[4] M. Haage, S. Schotz, and P. Nugues. A prototype robot speech interface with multimodal feedback. In *In IEEE ROMAN 2002, Proceedings of the 11th International Workshop on Robot and Human Interactive Communication*, pages 247–252, Berlin, September 2002.

[5] J. Hagg, B. Akan, B. Curuklu, and L. Asplund. Gesture recognition using evolution strategy neural network. Emerging Technologies on Factory Automation 2008, September 2008.

[6] http://www.microsoft.com/speech. *Microsoft Speech Technologies*.

[7] R. Kurnia, A. Hossain, A. Nakamura, and Y. Kuno. Object recognition through human robot interaction by speech. In *Procedings of the 2004 IEEE International Workshop on Robot and Human Interactive Communication*, pages 619–624, Okoyama, Japan, 2004.

[8] B. Landau and R. Jackendoff. What and where in spatial language and spatial cognition. *Behavioral and Brain Sciences*, 16:217–265, 1993.

[9] J. O. Peter Gorniak and D. Roy. Speech, space and purpose: Situated language understanding in computer games. Twenty-eighth Annual Meeting of the Cognitive Science Society Workshop on Computer Games, 2006.

[10] M. Skubic, D. Perzanowski, A. Schultz, and W. Adams. Using spatial language in a human-robot dialog. In *IEEE 2002 International Conference on Robotics and Automation*, pages 41–43, 2002.

[11] O. Sugiyama, T. Kanda, M. Imai, H. Ishiguro, N. Hagita, and Y. Anzai. Humanlike conversation with gestures and verbal cues based on a three-layer attention-drawing model. *Connect. Sci.*, 18(4):379–402, 2006.

[12] S. Tellex and D. Roy. Spatial routines for a simulated speech-controlled vehicle. In *In Human-Robot Interaction (HRI)*, March 2006.

[13] J. Wielemaker. An overview of the SWI-Prolog programming environment. In F. Mesnard and A. Serebenik, editors, *Proceedings of the 13th International Workshop on Logic Programming Environments*, pages 1–16, Heverlee, Belgium, december 2003. Katholieke Universiteit Leuven. CW 371.