A Virtual Robotic Agent that learns Natural Language Commands

J. Kontos, I. Malagardi & M. Bouligaraki
Artificial Intelligence Group
Laboratory of Cognitive Science
Department of Philosophy and History of Science
National and Capodistrian University of Athens
E-mail: ikontos@compulink.gr, imal@gsrt.gr

Abstract
The present paper presents the design and implementation of a Virtual Robotic Agent that learns Natural Language commands. The agent is a motion command understanding system with a machine learning interface for the communication with its user and teacher. The machine learning interface is based on two methodologies that exhibit two basic modes of learning. These two modes are "learning by example" and "learning by doing". The system described which consists of the agent and an I/O subsystem accepts Greek and English as the natural language of communication of the user with the agent and displays images to the user.
The execution of motion commands expressed in natural language is displayed graphically to the user so that he can easily follow the actions of the agent and supervise its learning operation. The system is applied to the study of the communication between a user and a virtual robotic agent, which exists in a virtual environment and accepts commands and knowledge about the objects and the actions possible in this environment. The commands express three kinds of actions. The first kind of action is change of position such as the movement of an object, the second kind is change of state such as the opening or closing of some objects and the third kind is the change of a relation between objects such as the placement of an object on top or inside another object. The two methods of machine learning used by the agent are presented using illustrative scenarios of operation of the agent.

Keywords:
 machine learning, natural language commands, virtual robotic agent

1. Introduction

The present paper presents the design and implementation of a virtual robotic agent with a command understanding and a learning interface for the communication with its user. The system is an improved version of the learning human-robot system as presented in [Kontos (1998)], [Kontos (1999)]. The machine learning interface is based on two methodologies that exhibit two basic modes of learning. These two modes are "learning by example" and "learning by doing". The two methods of machine learning used by the agent are presented using illustrative scenarios of operation of the agent.

The system described here accepts Greek and English as the natural language of communication of the user with the agent and the execution of motion commands expressed in natural language. The system could be applied to the communication between a user and the artificial agent, which acts in a virtual environment and accepts commands and knowledge about the objects and the actions possible in this environment.

The natural language commands that the virtual agent can execute refer to three kinds of actions. The first kind of action is change of position e.g. the movement of an object, the second kind is change of state of an object e.g. the opening or closing of some objects and the third kind is the change of a relation between objects e.g. the placement of an object on top or inside another object.

When the agent is given a command like: "open the door", "open the bottle", "close the box", "put the book on the desk" etc. which specifies a task, the agent has "to understand" the command before attempting to perform the task. Understanding such commands requires understanding the meaning of actions such as "open", "close", "put" and the meaning of prepositional words such as "on". The meanings of the constituents must be combined and the meaning of the sentence as a whole must be obtained taking into consideration knowledge of the environment or "microcosm" where the

commands must be performed. The execution of a command by the agent may initially be wrong. The interaction of the agent with the human user results in learning by the agent the correct way of executing a given command. This learning takes place in cases when the agent faces the problem of unknown words, of unknown senses of words or of under-specified positions of objects as well as when the user presents a new scenario to the agent that requires enrichment of its knowledge base .

The system was implemented with Turbo Prolog which has some simple facilities for computer graphics. Using these facilities the system displays a room with a door, some furniture and some mobile objects that can be manipulated by the agent. These are objects such as a table, a door, a desk, a bottle, a box and a book. The bottle, the box and the book are examples of mobile objects. The agent can move around in the room and execute the user's motion commands. These commands may refer directly or indirectly to the movement of specific objects or the change of their state. The agent knows the names of these objects and their position in the room displayed on the screen. The agent also knows how to execute some basic commands. Using the commands and the descriptions of the situation in the room the user can specify a scenario.

2. Motion Verbs

Motion may be specified by a verb either directly or indirectly. The simplest way to specify motion of an object is by using a verb that specifies motion directly. An example verb is "move" as used in the sentence "move the box". This sentence implies that a motion must be executed by the agent with the box as the affected object. Indirect specification of motion can be done in two ways: either in terms of geometric goals, or in terms of a force. Indirect specification of motion in terms of a goal involving physical relationship among objects is quite common. Consider the command "put the bottle on the table". This command requires that a physical object be moved i.e., "the bottle" with a goal to establish the physical relationship of "on" between it and another physical object i.e., "the table". Performance of such an instruction demonstrates that the goal of establishing a physical relationship drives the motion of the first object. For verbs such as "put" that specify motion in terms of a geometric goal, properties of the objects that participate in the underlying action are of crucial importance. Indirect specification of motion in terms of a force uses verbs such as "push" and "pull". Objects affected by motion commands may be also specified either directly or indirectly. Direct specification is based on names of objects known to the agent such as box, table, etc. Indirect specifications can be accomplished using complex noun phrases such as "the book on the desk".

In [Kalita (1997)] a fixed lexicon that is inserted manually was used. They state that their long -term goal is to investigate how semantic information for motion verbs can be automatically derived from machine readable dictionaries. They also state that at present their system has no feedback from the graphical animation system to the semantic processor. Finally their system has no learning capabilities. In contrast our system exhibits some novel features for treating motion verbs i.e. the creation of its lexicon is accomplished automatically using a machine readable dictionary, learning of the correct interpretation of commands with more than one meaning is accomplished using machine learning by supervision that are techniques based on visual feedback.

One source of the multiplicity of meaning of a command is the multiplicity of the senses of a word as recorded in a machine-readable dictionary. Another source is the possibility of an object to be placed on a surface in different ways. When the user submits a command, the agent, in order to satisfy the constraints of the verb may ask for new information and knowledge about objects and verbs, which may be used in the future. A machine-readable dictionary is used, which provides the definition of verbs. In particular, in the case of Greek about 600 motion verbs were analysed automatically in terms of about 50 basic verbs. Finally every time a command is executed which is amenable to more than one interpretation the system allows the user to observe the graphical output and state its approval or disapproval which helps the agent to learn by doing.

3. System Architecture and Operation

The system is composed of the components shown below.

Graphics Subsystem
Lexical Subsystem
  ➢ Machine readable dictionary processor
  ➢ Lexical processor
Syntactic Subsystem
  ➢ Chunker
  ➢ Syntactic processor
Semantic Subsystem
  ➢ Basic motion command processor
  ➢ Semantic processor
Learning Subsystem
  ➢ Learning by doing module
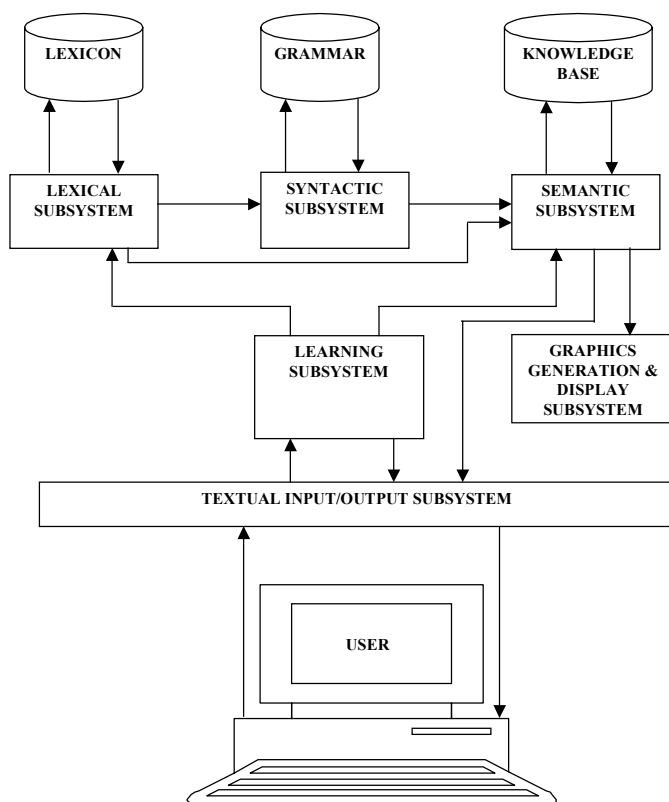 ➢ Learning by example module

Figure 1: Architecture of the Learning Agent System

The operation of these modules is supported by a number of databases. These are:

  ➢ Machine readable dictionary used offline
  ➢ Lexicon
  ➢ Grammar
  ➢ Knowledge Base

The graphical user interface that was implemented was very helpful during the development. It was easier to see the result on the screen, graphically rather than reading lists of the knowledge database to find the changes that were recorded during the program execution and the machine learning process.

The user enters his commands in natural language with the keyboard. The commands are imperative sentences with one motion verb that declares the action and some other words like nouns,

prepositions or adverbs complementing the main motion verbs. Prior to the syntactic and semantic analysis of the sentence the agent checks if each word of the sentence belongs to its lexicon. Stemming is used at this stage because of the complex morphology of the Greek words.

4. Agent Learning by Doing

When a command contains a word unknown to the agent then the system produces a message to the user asking for information about the unknown word and terminates the processing of the present command. After having recognized all the words in a command, the agent performs the syntactic analysis of it. If the input sentence is syntactically correct, the agent recognizes the motion verb, the object or objects and the adverb or preposition related to the verb. Then the module for "processing of motion commands" tries to satisfy all the constraints and the conditions for the specified motion. This processing requires searching in the knowledge base from where the agent retrieves information about the object's properties (e.g. position, weight, size, state, subparts etc.). At this point, when some information is unavailable or ambiguous, the agent interacts with the user in order to acquire the missing knowledge.

There are two different types of questions that the agent asks. The first type includes questions for which there is no information in the knowledge base and the user must supply it. The second type refers to questions which demand a Yes or No answer. This happens when more than one interpretation of an input command is possible and the agent cannot decide which is the correct one. In these cases, the system, using the machine learning mechanism, suggests each time one of the different solutions and requests an answer from the user. The Yes or No answers generate appropriate entries in the knowledge base and can be used next time a similar command is submitted by the user without requesting any more information.

Suppose that the user enters the command "open the door". The agent isolates the words of the command and recognizes the verb "open" and the noun phrase "the door". The verb "open" appears in the lexicon with a number of different definitions. E.g. in the LDOCE [LONGMAN (1992)] we find among others the senses of "open" a: to cause to become open, b: to make a passage by removing the things that are blocking it. The Greek dictionary we used contains similar sets of senses for this verb and the sense selection mechanism is practically the same for the two languages. The only difference is the wording of the sense selection rules for the two languages where the objects and their properties have different names. The agent selects the sense "b" because it knows that a door blocks a passage. The next decision the agent has to take concerns the way the opening action is executed.

The agent finds in the knowledge base that there are two alternative ways of interpreting the verb "open", using either a "push" or a "pull" basic motion. Then, it selects the first one and asks the user if this is the right one. If the answer is "No", the agent selects the next available interpretation and prompts again the user for an answer. When the answer is "Yes", a fact is recorded in the knowledge base which denotes that for the verb "open" and the object "door" the appropriate motion is e.g. "pull" in case that the "Yes" answer was given for the "pull" interpretation. The second example refers to the movement of a book that exists in the microcosm of the system. When the command "put the book on the desk" is given, the agent searches the knowledge database to find a side of the book that can be used as a base for it. The book has 6 sides and when the agent selects one of them it presents graphically the book on the desk having this side as base. Then, it asks the user if the result of the motion is the correct one. When the user enters a "Yes" answer, this is recorded in the knowledge base and the process terminates. When the user enters a "No" answer, the process continues trying sequentially all the available sides of the book until a "Yes" answer is given by the user.

5. Agent Learning by Example

Learning by example by the virtual agent is accomplished using the ID3 algorithm [Quinlan (1986)]. A Prolog implementation of ID3 algorithm was used for the second method of learning of the agent [Bouligaraki, (2000)]. The problem solved by ID3 is generation of a decision tree that on the basis of

answers to questions about the non-category attributes predicts correctly the value of the category attribute.

The computation of the entropy used to select the best attribute has been implemented with the following rule:

```
compute_set_entropy(Data,Entropy):-
count_positive(Data,num),length(Data,Dnum),
Pp=Pnum/Dnum, Pn=1-Pp,
xlogx(Pp,PpLogPp), xlogx(Pn, PnLogPn),
Temp=PpLogPp+PnLogPn, Entropy=-Temp.
```

Where Data is the input file and Entropy is the value of the entropy for the data. The predicate "count_positive" indicates the number of the examples that belong to the category being required using the rules:

```
count_positive([],0).
count_positive([dat("P",_)|More],Pnum):-
!,count_positive(More,Pnum1),Pnum=Pnum1+1.
count_positive([dat("N",_)|More],Pnum):-count_positive(More,Pnum).
```

The predicate "length" indicates the total of the examples using the rules:

```
length([],0).
length([Dat|Moredat], Dnum):- !, length(Moredat,Dnum1), Dnum=Dnum1+1.
```

The product "xlogx" is computed by the rules:

```
xlogx(X,N):-  X=0.0E+00,!, N=0.
xlogx(X,N):-  N=X*log(X).
```

The predicate select_minimal_entropy accepts a list of triples of the form:
```
(attribute,
partition_induced_by_that_attribute,
resulting_entropy)
```

and select the attribute that causes the partition with the least entropy and generates this partition given the structure "c=c(attr,partiton,entropy)" with the rules.

```
select_minimal_entropy([c(Attr,Partition,Entropy)|MorePartitions],Be
stAttr,BestPartition):-
select_minimal_entropy_aux(MorePartitions,c(Attr,Partition,Entropy),
BestAttr,BestPartition).
select_minimal_entropy_aux([],c(Attr,Partition,_),Attr,Partition).
select_minimal_entropy_aux([c(Attr1,Partition1,Entropy1)|
MorePartitions],
c(_,_,Entropy),BestAttr,BestPartition):- Entropy1<Entropy,!,
select_minimal_entropy_aux(MorePartitions,
c(Attr1,Partition1,Entropy1),BestAttr,BestPartition).
select_minimal_entropy_aux([_|MorePartitions],c(Attr,Partition,Entro
py),BestAttr,BestPartition):-
select_minimal_entropy_aux(MorePartitions,c(Attr,Partition,Entropy),
BestAttr,BestPartition).
```

In Figure 2, six example scenarios are shown from which the agent learns the decision tree shown in Figure 3.

EX	OBWEIGHT	OBSIZE	TARGSTRENGTH	TARGSIZE	ACTION
1	10	10	20	30	**YES**
2	20	10	20	30	**YES**
3	30	20	20	30	**NO**
4	40	20	30	40	**NO**
5	50	20	60	40	**YES**
6	60	30	50	40	**NO**

Figure 2: Training Data

OBWEIGHT TARGSTRENGTH

NO   YES

NO

NO   **OBSIZE TARGSIZE**

                        YES
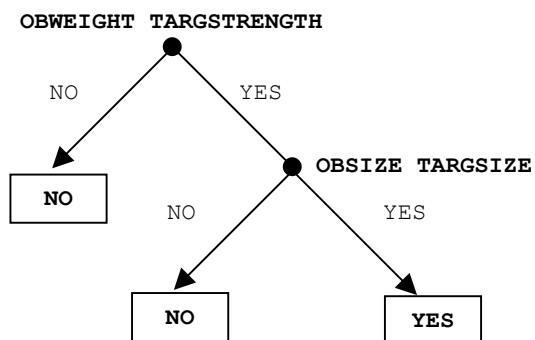
NO   **YES**

Figure 3: Decision Tree

6. Conclusions

A virtual robotic agent system was presented in the present paper. The learning module has two modes of operation namely learning by example using the ID3 algorithm and learning by doing using interaction with user. The agent is able to learn successfully how to obey natural language commands of moving objects in a room with some furniture taking into account the properties of the objects and the furniture. The dialog with the user is taking place in a friendly manner using a graphical display for monitoring the actions of the agent.

References

Bouligaraki, M. and Syllaios, G. (2000). Implementation of Machine Learning Techniques for Knowledge Extraction from Geographic/ Spatial Data, *2nd Conference – Information Systems in Agriculture.* Chania, Organisation: Greek Society for Operational Research – Technical University of Crete.

Kalita, J., K. and Lee, J., C. (1997). An Informal Analysis of Motion Verbs based on Physical Primitives.*Computational Intelligence,* Vol. 13, N.1, pp. 87-125.

Kontos, J. and Malagardi, I., and Trikkalidis, D. (1998). Natural Language Interface to an Agent. *EURISCON '98 Third European Robotics, Intelligent Systems & Control Conference* Athens. Published in *Conference Proceedings "Advances in Intelligent Systems:Concepts, Tools and Applications" (Kluwer)*

Kontos, J. and Malagardi, I. (1998). Question Answering and Information Extraction from Texts. *EURISCON '98 Third European Robotics, Intelligent Systems &Control Conference*. Athens. Published in *Conference Proceedings "Advances in Intelligent Systems: Concepts, Tools and Applications" (Kluwer)*. ch. 11,pp. 121-130.

Kontos, J. and Malagardi, I. (1999). A Learning Natural Language Interface to an Agent. *Proceedings of Workshops of Machine Learning ACCAI 99,* Chania. Crete. Hellas.

LONGMAN DICTIONARY OF CONTEMPORARY ENGLISH. (1978). *The up-to-date learning dictionary.* Editor-in-Chief Paul Procter. Longman group Ltd. UK.

Quinlan J., R. (1986). Induction of Decision Trees. *Machine Learning.* vol. 1, pp. 81-106.