Carl, a learning robot, serving food at the AAAI reception

L. Seabra Lopes

Transverse Activity on Intelligent Robotics
Departamento de Electrónica e Telecomunicações + Instituto de Engenharia Electrónica e Telemática
Universidade de Aveiro, P-3810 Aveiro - Portugal

Abstract

The development of robots able to accept, via a friendly interface, instructions in terms of the concepts familiar to the human user remains a challenge. Designing and building such intelligent robots can be seen as the problem of integrating four main dimensions: *human-robot communication*, *sensory motor skills and perception*, *decision-making capabilities* and *learning*. Although these dimensions have been thoroughly studied in the past, their integration has seldom been attempted in a systematic way. For the common user, a sufficiently practical interface must include spoken language. These issues are being studied at our institute in the framework of the project CARL ("Communication, Action, Reasoning and Learning in Robotics"). The *AAAI Hors d'Ouevres* competition seemed to be an interesting setting for evaluating and demonstrating the project's robot, naturally called 'Carl'. This paper describes the "body and soul" of Carl as well as the lessons learned from the participation in the competition.

1. Introduction

The development of robots that don't have to be programmed in the classical way and, instead, can accept instructions at the level of concepts of the human user will be a major breakthrough. If a flexible manufacturing system is supposed to produce a variety of products and in small quantities, then industrial robots will tend to play the role of craftsmen. Both service robots and flexible industrial robots will need to use sensors extensively in order to develop a high level understanding of their tasks.

Robot decision-making at the task level is, therefore, a central problem in the development of the next generation of robots. As the modularity and reconfigurability of the hardware are enhanced, the number of action alternatives at the task-level increases significantly, making autonomous decision-making even more necessary.

The development of task-level robot systems has long been a goal of robotics research (Lozano-Pérez et al., 1989; Borenstein and Koren, 1990). It is of crucial importance if robots are to become consumer products. The idea, that was already present in automatic robot programming languages since the 1970's, has been taken up in recent years by other researchers (Seabra Lopes, 1997).

The viewpoint in early artificial intelligence research was to evaluate an agent's intelligence by comparing it's thinking to human-level thinking. The development of human-level intelligence is probably a too ambitious goal for the current state of art. We believe that it is more reasonable to develop useful robotic systems with hardware and intelligence tailored for specific applications. This will provide experience on how to integrate different technologies and execution capabilities and, eventually, will enable us to scale up to more general robot architectures.

Currently, the major effort involved in developing useful intelligent robots is, we believe, in the integration of the different capabilities related to intelligence (Seabra Lopes and Connell, edrs., 2001).

The author is currently involved in a project titled "*Communication, Action, Reasoning and Learning in robotics*" (CARL).

CARL is based on the hypothesis that a combination of reactivity with reasoning is more likely to produce useful results in a relatively near future than the purely reactive or behavior-based approaches. This is especially true for robots that are expected to perform complex tasks requiring decision-making.

The integration of reactivity with reasoning has proved to be difficult to achieve. Traditional architectures have focused on traditional problems like reasoning, representation, and NLP and alternative architectures have focused on problems such as real-time perception and motor control. There have been few, if any, satisfying attempts to integrate the two. The position (and driving hope) of the CARL project is that most of the encountered difficulties are the result of not addressing properly the learning and, especially, the interface issues.

In the traditional approach to building intelligent systems, the human devises a formal language and uses it to specify the needed representations of the world. As the application becomes more and more complex, the programmer's task becomes overwhelmingly difficult. Automatic programming languages, embedding various planning capabilities, have been developed in order to simplify the programming problem. Programming by human demonstration and learning techniques (Morik et al., 1999) have been used for the same purpose. None of these approaches alone solved the problem. Robot programming is a bottleneck for robot development. The real underlying problem seems to be the grounding problem (Harnad, 1990).

To correctly address symbol grounding in the context of task execution, the first thing to notice is that most symbols are inherent to the tasks. In that case, the human user, who defines the tasks, will be a primary source of information for the symbol grounding process. The human will be simultaneously the user and the teacher. The communication interface between human and robot is, therefore, of primary importance.

If we are developing intelligent robots with significant decision making capabilities, the use of spoken natural language seems unavoidable. It seems unavoidable because no other alternative is practical enough for the common (naïve) user.

The paper is organized as follows. Section 2 describes the hardware configuration and software architecture of our robot, Carl. Section 3 describes the basic capabilities developed for Carl to support situated behavior and interaction. Section 4 describes the global management system of the robot. Section 5 describes the learning module. In the concluding section, some lessons learned from our participation in the *Hors d'Oeuvres* competition will be presented.

2. Carl, the robot

2.1. Hardware configuration

Carl is the name of the robot of the CARL project. It is based on a Pioneer 2-DX indoor platform from ActivMedia Robotics, with two drive wheels plus the caster. It includes wheel encoders, front and rear bumpers rings, front and rear sonar rings and audio I/O card. The platform configuration that was acquired also includes a micro-controller based on the Siemens C166 processor and an on-board computer based on a Pentium 266 MHz with PC104+ bus, 64 Mb of memory and a 3.2 Gb hard drive. The operating system is Linux. A Sony EVI pan-tilt camera was added.

On top of this mobile platform, we added a fiber glass structure that makes Carl approximately 85 cm high (see Fig. 1). This fiber structure carries a DA-400 v2 directional microphone from Andrea Electronics and a speaker. In a normal stand-up position near the robot, the mouth of a person is at a distance of 1 m from the microphone array. This is enough for enabling speech recognition in a quiet environment. This was, actually, the main motivation for adding the fiber structure: with the microphone installed directly in the Pioneer 2-DX base, the speech signal coming from a person in normal stand-up position would not be recognizable. For robust navigation, a set of 10 IR sensors was added to the fiber structure. The structure also includes a recipient for small objects, equipped with an IR sensor for detecting the presence of objects. In the *Hors d'Ouevres* competition, this was used for transporting food.

Fig. 1 – Current look of Carl

With this platform, we hope to be able develop a completely autonomous robot capable, not only of wandering around, but also of taking decisions, executing tasks and learning.

2.2. Software architecture

The control and deliberation architecture of Carl (Fig. 2) reflects the goals of our project. Human-robot communication is achieved through spoken language dialog. A set of Linux processes, making up the *speech processing* module, handle speech recognition, natural language parsing and speech synthesis. Another Linux process handles general perception and action, including navigation. High-level reasoning, including inductive and

deductive inference, is mostly based on the Prolog inference engine (we use a freeware implementation with a good C-language interface, SWI Prolog). Another module of the architecture provides Carl with learning capabilities. A central manager coordinates the activities at the high level. All these modules are described in special sections below.

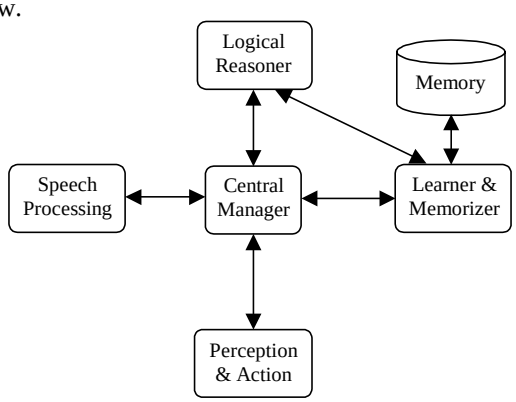


Fig. 2 – Software architecture of Carl – current form

3. Processes for situated activity

Carl is a prototype of a robot capable of performing actions in the real world according to spoken instructions from humans. Perception, navigation and spoken language processing are basic capabilities for such a robot.

3.1. Perception and navigation

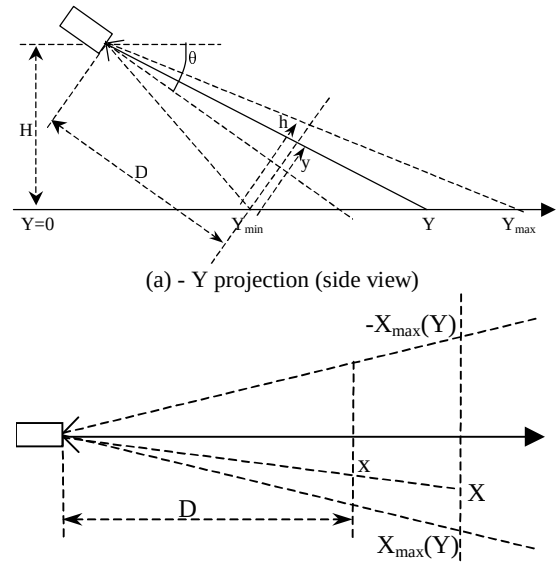
For robustly navigating in complex unstructured environments, such as an office or home environment, robots will benefit from using multiple sensor modalities. For instance, legs of chairs and tables are extremely hard to detect and locate by a robot only equipped with sonars.

The navigation strategy of Carl is based on the fusion of vision, sonar and infra-red sensing information. Vision information is used to build a local map of the robot's neighborhood. Since the only camera of Carl is the EVI camera, Carl does not perform any sort of stereo vision processing. Nevertheless, a single camera can be used to detect free space on the floor in front of the robot and, actually, build a top-view map of that free space, provided that the floor color is approximately constant in the robot's environment.

Fig. 3 illustrates the computation of the projection of an (X,Y) point on the floor to a plan parallel to the camera. Given the variables identified in the figure, the projection is computed by the following formulas:

$$x = \frac{D}{H} \cdot X \cdot \sin\left(\tan^{-1}\left(\frac{H}{Y}\right)\right)$$

$$y = D \cdot \tan\left(\theta - \tan^{-1}\left(\frac{H}{Y}\right)\right) + \frac{h}{2}$$

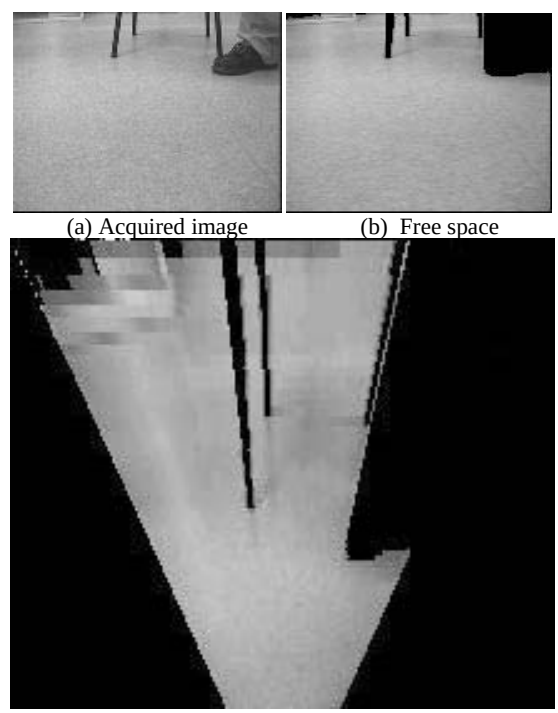


(a) - Y projection (side view)

(b) - X projection

Fig. 3 - Meaning of main geometric variables involved in the generation of a map of free space based on vision information

Based on this, the local map is built and maintained. By local map it is meant simply a structure storing the coordinates of points that represent, with a certain



(a) Acquired image (b) Free space

(c) Top view of free space in the visible area

Fig. 4 - Example of the process of generating a top view of free space on the floor in front of the robot

resolution (e.g 30 mm), the boundary of the free space around the robot. The process is as follows:

1. An image of the scene in front of the robot is captured. (e.g. Fig. 3a.)
2. Free space in the image is detected - this is done by scanning the image from bottom to top until, in each column, a pixel is found out of the intensity interval of the floor; this is considered as the border of an obstacle. (In Fig. 3b, occupied space is marked black; note that the foot and the chair legs are easily detected.)
3. Based on the above equations, a top view of free space is generated (Fig. 3c). Note that only the base of each obstacle (laying on the floor) is correctly located; this is enough for obstacle avoidance, since the base of the obstacle appears to be closer to the robot Whatever appears to be further way can be ignored.
4. The top view image is segmented into a grid; scanning this grid from bottom to top, the first cells that are found occupied (the average pixel intensity is on average below or above the floor intensity range) can be marked as potential obstacles in the local map.
5. As the robot moves around, the positions of these points relative to the robot are updated and those that are too far (e.g. more than 1.2 m) are removed; previously recorded points that are again in the view field of the robot are also removed, so that new points, corresponding to the current perception, can be recorded.

The top view image is also used to update a global map. The global map of Carl is currently a grid-based map. The grid resolution is 100 mm. Each time the robot sees that a given cell of the global map is free, according to the mentioned top view, an occupancy indicator for that cell is incremented. However, the map building and path planning capabilities of Carl were not demonstrated in the AAAI Competition.

During navigation, each obstacle point in the local map exerts a certain virtual force in the speed and direction of movement of the robot. The approach is based on the Virtual Force Field (VFF) concept of Borenstein and Koren (1991).

Provided that contrast between floor intensity and obstacle intensity is sufficient, Carl can navigate based on vision only. However, sometimes an obstacle can really be confused with the floor. Therefore, complementary sources of information are taken into account, namely sonar and infra-red sensing information.

Obstacles "seen" by sonars and infra-red sensors are also handled through the same VFF-like approach. The speed and angle values obtained by applying VFF to vision, sonar and infra-red data are combined to produce values that are finally applied to adjust the robot's trajectory.

This way, Carl is able to robustly navigate in complex human environments. In this sense, the environment that was set up for the AAAI reception was quite easy, since it contained almost no furniture.

3.2. Spoken language processing

A spoken language interface enables humans to comfortably instruct their robots. The spoken language processing modules address the well known problems presented in Table I. The current interface of Carl builds on work described in previous papers of our group (Seabra Lopes and Teixeira, 2000).

In this project, human-robot communication is modeled as the exchange of messages, much like is done in multi-agent systems. Se set of performatives or message types in our Human-Robot Communication Language (HRCL) is inspired in KQML, the outer language of ACL (Labrou and Finn, 1997). Table II lists the currently supported performatives.

For spoken language input, a grammar for a subset of the English language has been specified using the APSG (Augmented Phrase Structure Grammar) formalism. For each performative, a certain number of grammar rules has been written. In total, approximately 50 phrase structure rules are being used together with a vocabulary of approximately 100 words. This allows the grammar to accept over 12000 different sentences.

A set of public domain tools is being used by the project for spoken language processing. Speech recognition and speech synthesis are handled by Linux processes based on IBM ViaVoice. Natural language parsing and phrase structure construction are handled by another Linux process based on the CPK NLP suite (Brondsted,1999).

One of the problems of current spoken language systems is the lack of robustness of the speech recognition process. Variations in environment noise, speaker language accent or speaker tone of voice, have dramatic consequences on the recognition performance. For our experiments, HMM speech models (of ViaVoice) have been trained for a set of

Table I – Spoken language processing sub-problems

	Speech	Language	Semantics
Input	Speech recognition	Language parsing	Semantics extraction
Output	Speech synthesis	Language generation	Semantics construction

Table II – Currently supported performatives (S=sender, R=receiver)

Performative	Description
Register(S,R)	S announces its presence to R
Achieve(S,R,C)	S asks R to perform action C in its physical environment
Tell(S,R,C)	S tells R that sentence C is true
Ask(S,R,C)	S asks R to provide one instantiation of sentence C
Ask_if(S,R,C)	S wants to know if R thinks sentence C is true
Thanks(S,R)	S expresses gratitude to R
Bye(S,R)	S says good-bye to R
Dye(S,R)	S (human master) asks R (robot) to close execution processes

```
semantics(
  tell(phrase(NP1,verb(be),prep(P),NP2)),
  is_(NP1sem,What),
  L3
):- semantics(NP1,NP1sem,L1),
    semantics(NP2,NP2sem,L2),
    What =.. [P,NP2sem],
    append(L1,L2,L3).
```

Fig. 5 - A semantics extraction rule

four speakers. With this training and in a reasonably silent environment, Carl is able to recognize utterances well enough to enable dialogue.

Of course, the utterance must be acceptable by the grammar. The large grammar that is being used stretches the limits of current technology. However, large as it appears to be, it still covers only a small part of the English language. For instance, in the tell, ask and ask_if performatives, only sentences based on the verb *to be* are accepted, for example: "*The professor is in Portugal*"; "*The car of Peter is at the University*"; or "*The chairman of the conference is a professor*".

The final step in processing an utterance is the extraction of the semantics from the phrase structure description produced by the CPK parser. This is done by a Prolog program designed and implemented by our group. The semantics of a sentence is a relational description. For instance, the Prolog clause given in Fig. 5 extracts the semantics of sentences that are based on the verb *to be* and include a prepositional phrase. A recursive call extracts the semantics of the noun phrase, producing NP1sem and additional relations in list L1. A similar call handles the other noun phrase. Finally, the semantics is given by the relation is_(NP1sem,What), with other complementary relations given in list L3. Many other clauses of this type handle the different cases allowed by the grammar. As an example, the semantics of "*Professor Carlos is at the university of Aveiro*", would be represented by the following list of relations, as computed by the program: [is_(X, at(Y)), is_(X, professor), obj_name_(X, carlos), of_(Y, Z), is_(Y, university), obj_name_(Z, aveiro)].

4. Execution management

The central manager is an event-driven system. Events originating in the speech interface, in sensors or in the navigation activity as well as timeout events lead to state transitions. Such apparently different activities as dialog management and navigation management are integrated in a common unified framework.

```
state_transition(
  State,
  [no_biscuits],
  ( member(State,[explore,wander,stay]) ),
  nothing,
  [ retract_all_times,assert_go_to_refill_time,
    execute_task(go_to_refill_area) ],
  going_to_refill
).

state_transition(
  interacting,
  [ heard(tell(Phrase)) ],
  true,  % no restrictions
  acknowledge_told_fact(Phrase),
  [ execute_motion(stop),retract_all_times,
    memorize_told_fact(Phrase),assert_last_heard_time],
  interacting
).
```

Fig. 7 – Examples of state transitions

It is mostly implemented in Prolog, in order to have easy access to the Prolog inference engine. Some parts of the manager are written in C language, either for reasons of efficiency or for access to the Unix inter-process communication facilities.

The central manager is essentially a state transition function (Fig. 6) specified as a set of Prolog clauses. Each clause, specifying a transition, has a head of the following form:

```
state_transition(State,Events,Restrictions,
                                SpeechAct,Actions,NewState)
```

State is the current state; Events is a list of events that will cause a transition to NewState, provided that the Restrictions are satisfied. These events can be speech input events, navigation events, timing events, robot body events. SpeechAct, if not void, is some verbal message that the robot should emit in this transition. Actions are a list of other actions that robot should perform. These can be

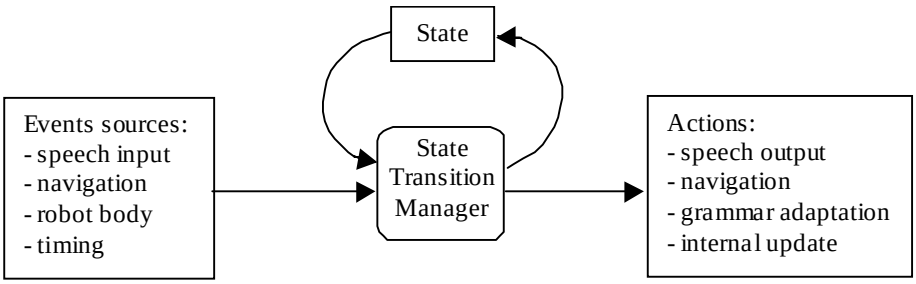


Fig. 6 - The central manager module - an event-driven process

actions related to navigation, but also internal state update and dynamic grammar adaptation.

Fig. 7 shows two examples of state transitions. The first one is a transition from a normal motion state (explore or wander) or stay state to a state in which the main activity of the robot is to go to the refill area. The triggering event is the absence of biscuits in the food tray of the robot. This activity, event and state transition were introduced for the AAAI competition. The second state transition in Fig. 7 is a transition to the same state, in this case the interacting state. The triggering event is the reception of an instance of the tell performative. The robot immediately stops and acknowledges, then memorizes the told information. The time of this event is recorded, so that the robot may later recognize that the interaction is over, if it didn't finish with an explicit "good bye" from the human interactant.

5. Learning

Learning and grounding are key concerns in our project, as already pointed out.

For a given robot, the idea is to integrate, in a so-called “construction phase”, a variety of processing and inference capabilities. In contrast, the initial body of knowledge should be minimal. After this phase is concluded (after the robot is born!), a life-long learning process can start. The robot learns new skills, explores its environment, builds a map of it, all this with frequent guidance from human interactants.

Some of the "innate" capabilities / knowledge, that should be integrated during the construction phase are:

1. Wandering around in the environment while avoiding obstacles; this would be the only "innate" physical behavior. (see developments in section 3.1)
2. Natural language processing (see section 3.2), supported by a fairly comprehensive vocabulary of English words; the meanings of most words are initially unknown to the robot.
3. Basic speech processing (section 3.2).
4. A small dictionary of words and their meanings for identifying the robot's sensors and basic movements; these are the initially ground symbols over which the robot will incrementally build its knowledge.
5. Ontologies for organizing and composing behaviors, map regions, dialogues, task plans, episodic memory, etc.
6. Knowledge of basic mathematical functions, that the teacher can use for teaching new concepts or behaviors.
7. Logical deduction (the Prolog engine is being used).
8. Capabilities for task planning and execution monitoring.
9. Capabilities for learning numerical functions.
10. Capabilities for learning symbolic classification knowledge.
11. Capabilities for explanation-based learning and case-based reasoning.

Part of these capabilities have already been integrated in the current Carl prototype. Others, related to explanation-based/case-based learning, will be supported in the near future, through the integration of modules previously developed by the research team (Seabra Lopes, 1997 and 1999ab).

It should also be noted that on-line lifelong learning in robotics has seldom been described. Moreover, the few known systems demonstrating on-line learning, still are mostly limited to sub-symbolic learning.

In the case of Carl, recent efforts in this field are mainly aimed at building something that demonstrates on-line symbolic learning. The perfection of the architecture is not a priority yet, as this is still groundbreaking work. Two main learning tasks are being addressed.

- learning facts about the world through interaction with humans
- on-line human-supervised learning for object recognition and symbol grounding

The architecture of the learning module is illustrated in Fig. 8. Semantic information extracted from tell messages received from the human interactant are stored in a database of logical assertions (actually, the Prolog database). Here is an example of a dialog that leads to learning:

[Learning a new fact:]
H – *Hello, Carl!*
C – *Hi, would you like some food?*
H – *Thank you!*
C – *You are welcome.*
H – *Professor Doty is in Portugal.*
C – *Ok.*
[Later, provide the learned information:]
H – *Where is the professor?*
C – *Portugal.*
[Or:]
H – *Is the professor in France?*
C – *No.*

Of course, this is learning of mostly non ground information. Nevertheless, this sort of functionality, if robust, may be useful for real-world applications.

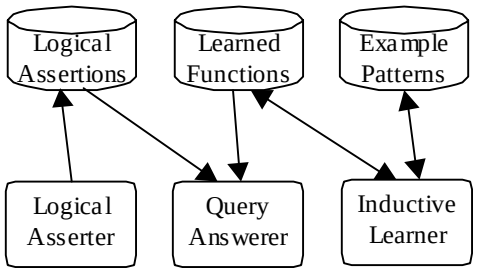


Fig. 8 - The learning and memorization module

The learning module of Carl also includes an inductive learner that we want to use for learning ground concepts. The particular task is the lifelong learning of object recognition knowledge. For instance, when Carl meets an obstacle, he may decide to ask:

"*Is this a person?*"

Based on the obtained answer ("yes" or "no") and the visual feedback, Carl may store a classified example. A collection of labeled examples like this, will enable a supervised learning algorithm to induce the concept of "person".

The inductive learner that was developed, based on plain backpropagation neural networks, allows for the concurrent learning of multiple concepts. It works as a learning server for the robot. Although our focus is now on object recognition, it can also be used for synthesizing behaviors based on data collected in training sessions conducted by human teachers (Seabra Lopes and Teixeira, 2000)

The complete cycle of lifelong learning from examples has not been demonstrated yet. Current efforts are concerned with automated feature extraction for improving the learning performance.

6. Lessons learned / Conclusion

Carl is a prototype robot that demonstrates the integration of communication, perception/action, reasoning and learning. It is an on-going project. Nevertheless, Carl is already able to: navigate in complex unstructured environments; enter in dialogue with a human being; and to learn some information from the human being. Current work is particularly concerned with the grounding problem.

The interesting evaluation opportunity provided by the *AAAI Hors d'Ouevres Competition*, in Seattle, has shown, in the first place, that it is extremely hard to have successful speech recognition in a crowdy and, therefore, noisy reception. During the reception, Carl almost didn't enter in dialog with humans due to the environment noise. It did say "*Hello, would you like some food?*", but almost never heard the reply, as it would, if the environment was quiet. This seems to show that it is still difficult to have talking robots being used in real-world applications.

Of course, this does not mean that the technology won't improve. Therefore, we will continue with our line of research on human-assisted learning, that we consider a basic ingredient for future intelligent robots.

From the point of view of navigation and collision avoidance, the AAAI reception was not particularly tuff. Although it was crowdy, humans are sufficiently large to be easily detected and avoided by robots. Carl would be able deal with more complex environments.

Acknowledgements

CARL is funded by the Portuguese research foundation (FCT), program PRAXIS XXI, reference PRAXIS/P/EEI/12121/1998. Implementation of the different modules of the Carl software architecture was maily carried out by the author and students Mário Rodrigues, Nuno Alves and Gabriel Guerreiro. Special hardware for the infra-red sensing system was developed by students João Capucho, Nuno Alves and Gabriel Guerreiro. Enlighting discussions with and technical advice from colleagues António Teixeira, Armando Pinho, José Luís Azevedo and Bernardo Cunha were also much appreciated. Special thanks go to AAAI for supporting a substantial part of the travel costs involved in the participation of the CARL team in *11th AAAI Robot Competition*.

References

Borenstein, J. and Y. Koren (1990) "Task-Level Tour Plan Generation for Mobile Robots", *IEEE Transactions on Systems, Man, and Cybernetics*, 20 (4), pp. 938-943.

Borenstein, J. and Y. Koren (1991) "The Vector Field Histogram - Fast Obstacle Avoidance for Mobile Robots", *IEEE Journal of Robotics and Automation*, 7 (3), p. 278-288.

Brondsted, T. (1999) "The CPK NLP Suite for Spoken Language Understanding", *Proc. of Eurospeech'99*, Budapest, Hungary.

Harnad, S. (1990) «The Symbol Grounding Problem», *Physica D*, vol. 42, pp. 335-346.

Labrou, Y. and T. Finin (1997) *A Proposal for a New KQML Specification*, University of Maryland at Baltimore County, technical report CS-97-03.

Lozano-Pérez, T., J.L. Jones, E. Mazer and P.A. O'Donnell (1989) "Task-level Planning of Pick and Place Robot Motions", *Computer*, vol. 22, n.3 (March), pp. 21-29.

Morik, K., M. Kaiser and V. Klingspor, edrs. (1999) *Making Robots Smart. Behavioral Learning Combines Sensing and Action*, Kluwer Acedemic Publishers.

Seabra Lopes, L. (1997) *Robot Learning at the Task Level: A Study in the Assembly Domain*, Universidade Nova de Lisboa, Ph.D. Thesis.

Seabra Lopes, L. (1999b) «Failure Recovery Planning in Assembly Based on Acquired Experience: Learning by Analogy», *Proc. IEEE. Int. Symp. on Assembly and Task Planning*, Porto, Portugal.

Seabra Lopes, L. and J.H. Connell, edrs. (2001) *Semi-Sentient Robots* (special issue of *IEEE Intelligent Systems*, vol. 16, n. 5), Computer Society, p. 10-14.

Seabra Lopes, L. and A.J.S. Teixeira (2000) Human-Robot Interaction through Spoken Language Dialogue, *Proceedings IEEE/RSJ International Conference on Intelligent Robots and Systems*, Japan.