From Reactive to Cognitive Agents: Extending Reinforcement Learning to Generate Symbolic Knowledge Bases

Rômulo G. Cerqueira
and Augusto Loureiro da Costa
Robotics Laboratory
Federal University of Bahia
Salvador, Bahia, Brazil 40210-630
Emails: romulogcerqueira@gmail.com,
augusto.loureiro@ufba.br

Stephen G. McGill
and Daniel Lee
and George Pappas
GRAP Lab
University of Pennsylvania
Philadelphia, Pennsylvania, USA
Emails: smcgill3@seas.upenn.edu,
ddlee@seas.upenn.edu,
pappasg@seas.upenn.edu

***Abstract*—A new methodology for knowledge-based agents to learn from interactions with their environment is presented in this paper. This approach combines Reinforcement Learning and Knowledge-Based Systems. A Q-Learning algorithm obtains the optimal policy, which is automatically coded into a symbolic rule base, using first-order logic as knowledge representation formalism. The knowledge base was embedded in an omnidirectional mobile robot, making it able to navigate autonomously in unpredictable environments with obstacles using the same knowledge base. Additionally, a method of space abstraction based in human reasoning was formalized to reduce the number of complex environment states and to accelerate the learning. The experimental results of autonomous navigation executed by the real robot are also presented here.**

I. Introduction

In recent years, the academic community of Artificial Intelligence has searched to replace explicit programming by the process of teaching a task. Autonomous robot skill acquisitions have presented a large amount of works using learning processes. In this context, Reinforcement Learning (RL) is a widely used technique, where an agent learns a policy π, exploring a partially or totally unknown environment, based in rewards obtained from its actions. Thus, the optimal policy π^* maps each environment state s_t to the best action a_t and it is stored in a hash table.

In task execution, this method presents the classical reactive agents limitations, some situation, like when the robot is surrounded by mobile obstacles, it collapses. Besides, the machine learning process usually takes a long time to learn a new desired behaviour. New learning episodes also are needed for each new environment configurations. In unpredictable changing environments with unpredictable obstacles, the agent can spend more time learning than executing the desired task. Moreover, the storage and management of several look-up tables becomes hard work for a reactive agent.

A two layered approach, where basic behaviours are implemented at a low level layer and supervisory fuzzy rule based system is implemented at a high level layer, decides the appropriate basic behaviours in the current environment state is presented in [2]. The fuzzy rules base was automatically generated by Reinforcement Learning algorithm. This improves the behaviour based navigation idea in [3], and extends the Reinforcement Learning to generates a fuzzy rules based system. In another approach [12], Inductive Logic Programming is used to transfer learning, where relevant knowledge from previous learning experiences was applied to help the new task learning. Recently, a significant amount of interesting research works about mobile robot navigation has been presented. In these works, some widely utilized methods are neural networks in [9], fuzzy logic in [11], reinforcement learning in [4], genetic algorithms in [5], and rule-based-neuro-fuzzy technique in [10].

Looking forward to give full decision autonomy to mobile robots, in high level tasks executions, a cognitive agent called Concurrent Autonomous Agent architecture (CAA) [7], was chosen to be embedded in the respective mobile robot. This agent architecture, embodied a model for higher-level cognition that supports learning and reasoning mechanisms using logical expressions. The cognitive model is based on the following fundamental hypotheses [6]:

- Cognition is an emergent property of a cyclic dynamic self-organizing process based on the interaction of a large number of functionally independent units of a few types.
- Any model of the cognitive activity should be epistemologically compatible with the Theory of Evolution [8]. That applies not only to the "hardware" components of this activity but also to its "psychological" aspects.
- Learning and cognitive activity are closely related and, therefore, the cognitive modelling process should strongly depend on the cognitive agents particular history [6].

Since learning and cognitive activity are closely related, the methodology that combines RL and Knowledge-Based Systems, was implemented to allow the CAA learns with its

978-0-7695-5139-5/13 $26.00 © 2013 IEEE
DOI 10.1109/.37

interaction with the environment. This approach makes CAA fully compatible with the three hypothesis for a cognitive model. In the previous CAA implementation, the learning propriety would be eligible just for reactive level, since that both instinctive and cognitive level uses Knowledge Based Systems for automatic reasoning. The methodology was applied to the instinctive level, where a knowledge-based system is used to plan execution. For a given goal g_i, the RL obtains the optimal policy which is automatically coded using a knowledge representation formalism based on first order logic. Each knowledge base KB_i stores the necessary knowledge for the agent to achieve the goal g_i, combining a set of basic behaviours, that can be performed in the environment. A new approach for environment mapping around the mobile robot neighbourhood allows a significant reduction of the relevant state space.

The paper is organized as follows: In Section II, the proposed approach for abstraction of mobile robot neighbourhood is presented. Section III describes the algorithm based on RL, where the optimal policy π^* is learned, and then coded into the knowledge bases KB_i. Section IV, presents experimental results from simulated mobile robot locomotion, and real mobile robot locomotion. Section V discusses conclusions and future works.

II. Reduced States Space Mapping

When a human tries to walk safely in a dynamic environment, he will care for the relative or approximate distances and directions between him and the goal and the closest obstacle, navigating safely independently of the number of obstacles. In this work, the agent learns based on human reasoning. The robot was considered to be the center of universe and the environment surrounding was divided into four orthogonal regions: R1, R2, R3 and R4, as seen in figure 1.

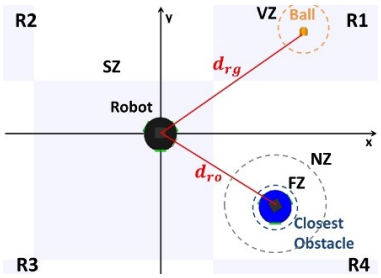

Fig. 1. The Environment Zones and States Space

The environment state at each time t is determined by the goal region (R_g) and closest obstacle region (R_o):

$$S_t = (R_g, R_o), t \in [1, 16] \quad (1)$$

This approach dramatically reduces the environment to 16 states, accelerates Q-Learning, decreases computational cost, and more specifically generates a small knowledge base, which is very appropriate for embedding in mobile robots.

This work uses global vision system, where the environment surrounding the robot is known, but could also use an omnidirectional local vision system. This works adopts some assumptions:

1) The robot's position and velocity are known at each instant;
2) The goal's position and velocity are known at each instant;
3) The closest obstacle's position and velocity are known at each instant.

The new methodology consists of three steps: agent learning, simulated environment execution, and embedded knowledge base and task execution into mobile robot. The first and second steps are done into a simulated environment. Once the optimal policy π^* is generated it is coded into a symbolic knowledge Base KB which is embedded into CAA's instinctive level. Then new simulation experiments, where the environment assumes different configurations, are performed. Finally the agent is embedded into the omnidirectional mobile robot, and the free collision navigation experiments are done. In a second experiment, now using a NaO humanoid robot, the agent reactive level was replaced by another version which embedded a Zero Moment Point Controller and the local Vision systems, provided by the Unified Humanoids Robotic Software Platform [13], and that instinctive level and mainly the Knowledge Bases was kept the same, and the same free collision navigation experiments were successful repeated with the humanoid robot.

III. Reinforcement Learning for Trajectory Planning

The RL model that CAA uses in this work, is presented in figure 2. At each time instant t (iteration), the agent uses its sensors to perceive the current environment's state s_t. Following a policy π, an action a_t is chosen and executed by its actuators. This action leads the environment to assume a new state s_{t+1}. The agent is rewarded if its taken action was good; otherwise it is punished. This reinforcement measurement, choice of the two works over time, will cause a correction of policy, defining a dynamic, interactive and learning process.

A π policy maps the environment states into actions, and defines the agent behaviour over time. Thus, the role of RL system is to find an optimal policy π^* that maximizes the reward R returned by environment. Once the optimal policy π^* is learned, it is coded in knowledge bases.

The proposed algorithm, implemented in C programming language, is based in Q-Learning, a machine learning technique that has been successfully utilized for solving the mobile robot navigation problem in dynamic environments.

A. The Set of Actions

An omnidirectional robot and robot soccer field, with Robocup F180 League official dimensions, composes the environment in this work. According to the holonomic constraints of an omnidirectional robot, the set of actions is defined by eight movement actions:

North, Northeast, East, Southeast, South, Southwest, West, Northwest.

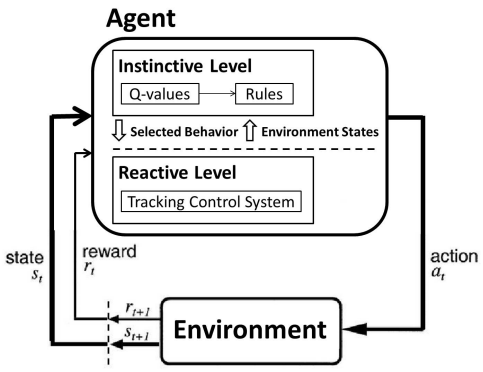

Fig. 2. RL model with CAA architecture

B. The Reward Function

The reward function is defined by the euclidean distances between the robot and goal (d_{rg}) and closest obstacle (d_{ro}). Thus, the set of states is classified as follows:

- Safe Zone (SZ), when the robot has no or low possibility of collision with any obstacle;
- Not-Safe Zone (NZ), when the robot has high possibility of collision with any obstacle;
- Victory Zone (VZ), when the robot reaches its goal (the ball in this case);
- Failure Zone (FZ), when the robot collides with an obstacle.

Moreover:

- rad_{vz} is the radius of VZ around the goal;
- rad_{nz} is the radius of NZ around the closest obstacle;
- rad_{col} is the radius around closest obstacle.

The robot zone at time instant t (figure 1), is defined as:

$$Z(t) = \begin{cases} VZ, d_{rg} \leqslant rad_{vz} \\ FZ, d_{ro} \leqslant rad_{col} \\ SZ, d_{ro} > rad_{nz} \\ NZ, rad_{col} < d_{ro} \leqslant rad_{nz} \end{cases} \quad (2)$$

From data, the reward function in time instant t is defined:

$$r = \begin{cases} 4, Z \subset SZ \rightarrow VZ \\ 4, Z \subset NZ \rightarrow VZ \\ 1, Z \subset NZ \rightarrow SZ \\ -1, Z \subset SZ \rightarrow NZ \\ 3, Z \subset NZ \rightarrow NZ, d_{ro}(t+1) < d_{ro}(t), \\ \quad d_{rg}(t+1) < d_{rg}(t), d_{ro}(t+1) \geq d_{rb}(t+1) \\ -3, Z \subset NZ \rightarrow NZ, d_{ro}(t+1) < d_{ro}(t), \\ \quad d_{rg}(t+1) < d_{rg}(t), d_{ro}(t+1) < d_{rb}(t+1) \\ -2, Z \subset NZ \rightarrow NZ, d_{ro}(t+1) < d_{ro}(t), \\ \quad d_{rg}(t+1) \geq d_{rg}(t) \\ 2, Z \subset NZ \rightarrow NZ, d_{ro}(t+1) \geq d_{ro}(t), \\ \quad d_{rg}(t+1) < d_{rg}(t) \\ -1, Z \subset NZ \rightarrow NZ, d_{ro}(t+1) \geq d_{ro}(t), \\ \quad d_{rg}(t+1) \geq d_{rg}(t) \\ 0, Z \subset SZ \rightarrow SZ \\ -4, Z \subset NZ \rightarrow FZ \end{cases} \quad (3)$$

C. The Value Function

Major RL algorithms are based on value functions. Q-Learning stores the value function in a Q-table, initially null. During the learning process, Q-table is filled by seeing different scenarios. This table has an immediate reward value from the taken action and the maximum accumulated reward for the possible actions in a current state (see Equation 4).

$$Q(s_t, a_t) = r(s_t, a_t) + \gamma * max(Q, s_{t+1}) \quad (4)$$

where: s_t is the state at time instant t; a_t is the action taken at t; $r(s_t, a_t)$ is the immediate reward received when an action a_t in a state s_t is taken; γ is the reduction factor, a number in the range of [0,1] that determines if the learned policy will consider delayed or immediate rewards, and $max(Q, s_{t+1})$ is the maximum Q-value calculated for taking all possible actions on the state at the next time step.

D. Learning Process

The learning process is composed by a given number of episodes. For each episode, one give scenario is presented by the agent that chooses an action, considering the goal and the expected reward. Q-table is updated by equation 4. Each scenario consists of the following: the robot position, the desired position, and just one static obstacle. Since the robot only considers with the nearest obstacle, then one obstacle is enough for training.

The element positions are chosen randomly for each episode. The robot always moves toward the goal, independent of a possible collision. It allows the Q-table to converge to the optimal policy π^*, and the robot remains in SZ until it reaches the desired destination.

Once inside SZ, the agent move towards the goal. If the last action lead to a state transition to a VZ or FZ, the episode ends. Thus, the Q-table is updated only when the agent is inside NZ. If it is the last episode, learning is finalized. The learning process is presented in figure 3.

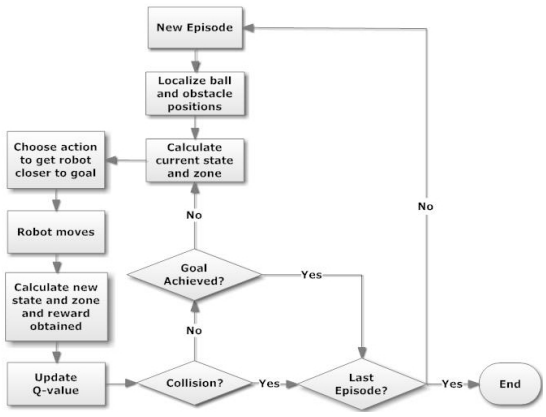

Fig. 3. Learning Process

At the end of learning, an algorithm written in C language converts automatically the optimal policy presented by Q-table into a rule base KB. Given the set of environment

states S and the set of possible actions A, the biggest Q-value for each state presents the best action to be executed by agent. For these experiments, six rules base KB where generated, which contains the optimal policies, for 10,000, 25,000, 50,000, 100,000, 250,000 and 500,000 episodes. Some of the 19 rules contained in KB, are presented on Table 1.

```
(rule_1
   (if (logic (robot zone not_safe))
       (logic (target quadrant r1))
       (logic (obstacle quadrant r1)))
   (then (logic (robo action east)))(cf 0.84))
                          ...
(rule_17
   (if (logic (robot zone safe)))
       (logic (target direction ?x))
   (then (logic (robot action ?x)))(cf 0.90))

(rule_18
   (if (logic (robot zone victory)))
   (then (logic (robot action success)))(cf 0.83))

(rule_19
   (if (logic (robot zone colision)))
   (then (logic (robot action stop))))(cf 0.85))
```

TABLE I. RULES BASE OF MOBILE ROBOT NAVIGATION

From rule 1 to rule 17, the rules chooses the basic behaviour according to the robot zone, the goal position, and the closest obstacle position. Rules 18 and 19, both require the reactive level to stop the robot and inform the cognitive level about the rules conclusion. Rule 18 identifies that the current goal was satisfied. In this situation, another goal should be chosen by the cognitive level.

Also, for rule 19, which characterizes a state where the mobile robot is surrounded by mobile obstacles, the cognitive level chooses another goal. This choice implies that another knowledge base must be used to scape from surrounded state, or wait for the collision zone state became false, it means wait for an mobile obstacles move out. The received reward for an action of the optimal policy π^* is normalized and stored in the rule as the certainty factor for that rule. A file in text format is generated which contains the rules base, or in other words the knowledge to execute the human like locomotion for the desired goal g_i. This file is downloaded to the omnidirectional mobile robot where the agent uses the stored knowledge to control the mobile robot locomotion.

IV. EXPERIMENTAL RESULTS

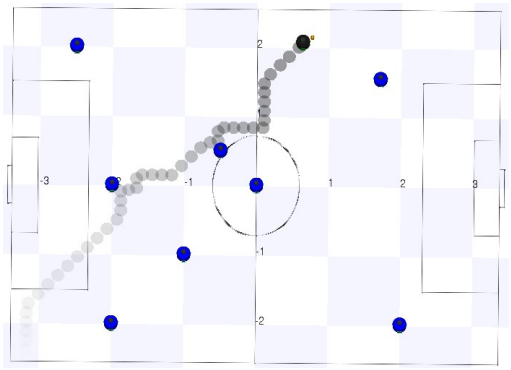

Fig. 4. A success path execution with obstacles

Once the symbolic knowledge base was embedded into CAA, two kinds of mobile robot locomotion experiments were done. First with the omnidirectional mobile robot AxeBot, and than with the NaO Humanoid robot. In both experiments, a simulated environment was used at first, and then with real mobile robot. The first experiment, the simulated one, the execution scenario was composed by: the robot, one target, and from 3 (three) to 8 (eight) obstacles (static and dynamic). The positions of the environmental elements are randomly chosen, including the ball that is the target, or the goal. One of these scenarios, where 8 (eight) obstacles are used, is shown at figure 4.

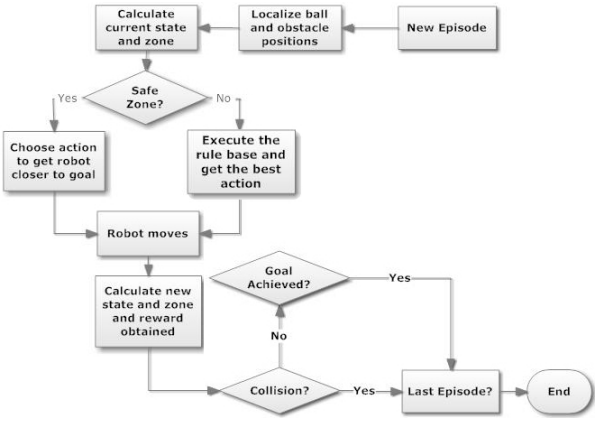

Fig. 5. Testing Process

For each symbolic knowledge base, KB_i, 500 (five hundred) scenarios, which contain, from 3 (tree) to 9 (nine) obstacles, and the ball, randomly positioned, were tested. The effectiveness of each different policy automatically coded into the symbolic knowledge bases is presented at **??**. The simulation algorithm is seen in figure 5.

Clearly, the knowledge base generated from the 500,000 iterations presents, award majority, a better effectiveness then the other ones.

The second step seeks to validate the developed methodology for a mobile robot. The file which contains a knowledge base, generated by the 500,000 iteration learning process, was loaded into the CAA instinctive level embedded on the omnidirectional mobile robot. This file stores the motion plan, combining basic reactive behaviours, to move the robot to the current ball position, which contains 19 rules. Three different scenario, where used for this experiment. The respective trajectory tracking for one of these scenarios and the temporal evolution of robot positions can be seen at figure 6. The path was correctly planned and the real robot was able to track it under the real-time constraints, support the proposed methodology.

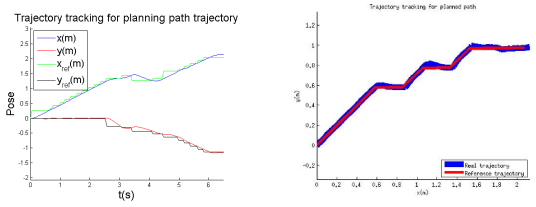

Fig. 6. Temporal Evolution of Positions and Mobile Robot Trajectory

In the second experiment a new reactive level, which embedded a Zero Moment Point (ZMP) motion controller for humanoid robots was implemented. It was encapsulated in a process which communicates with the instinctive level using udp sockets. This new reactive process also encapsulates the local vision system, see Figure 7. The same exchange message pattern between the reactive level and the instinctive level was kept. The reactive level sends to instinctive

level, the robot pose, the target pose and the position of the seen obstacles. The instinctive level identifies the current state and choose the behaviour that is executed by the ZMP motion controller.

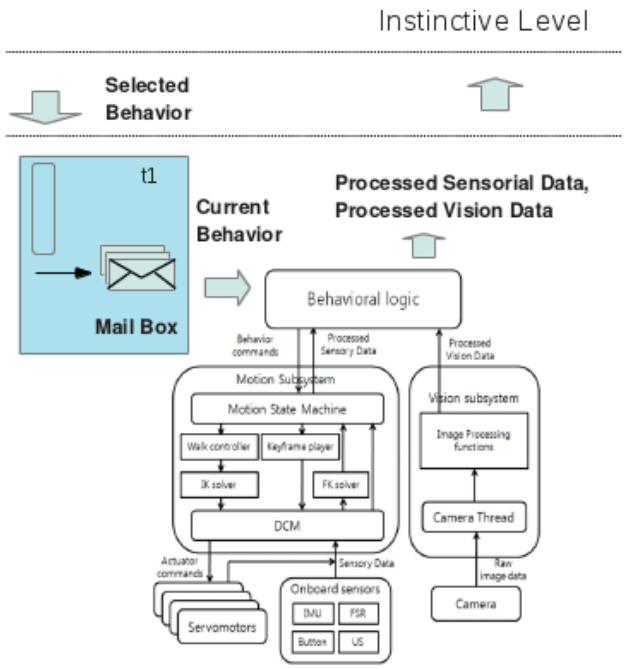

Fig. 7. The Reactive Level for NaO Humanoid Robots

The same Three different scenario, used at the first experiment with the omnidirectional mobile robot were successful repeated. The respective trajectory tracking for one of these scenarios and the temporal evolution of robot positions can be seen at Figure 8. Again, the path was correctly planned and the real robot was able to track it.

V. Conclusions and Future Works

A new methodology for knowledge-based agent to learn the knowledge from their interactions with the environment is presented in this paper. The new methodology combines Reinforcement Learning and Knowledge-Based Systems. The Reinforcement Learning obtains the optimal policy which is automatically coded using a knowledge representation formalism based on first order logic. The generated knowledge base is used by a cognitive agent embedded into the omnidirectional mobile robot that executes the learned human like locomotion task.

Considering only the closest obstacle, the state space of the environment is reduced drastically which accelerates the convergence to the optimal policy and mainly allows the proposed methodology to generate small knowledge bases appropriate to be embedded into mobile robots. The experimental tests realized show the planned points in real-time by the Instinctive level were received and executed successfully by the Reactive level.

For a different goal g_j, for example dribbling around an opponent, a new learning process would be necessary. This new learning process would generate a new knowledge base KB_j, which, once added to mobile robot, would make it able to achieve both g_i and g_j. In this sense, where a specific knowledge base is generated for a given task, it is possible see the proposed methodology as a symbolic knowledge learning methodology. Of course a good discussion about symbolic knowledge learning can be further researched in this situation, and we hope to support it with future additional results where the robot increases its skill with additional learned knowledge bases.

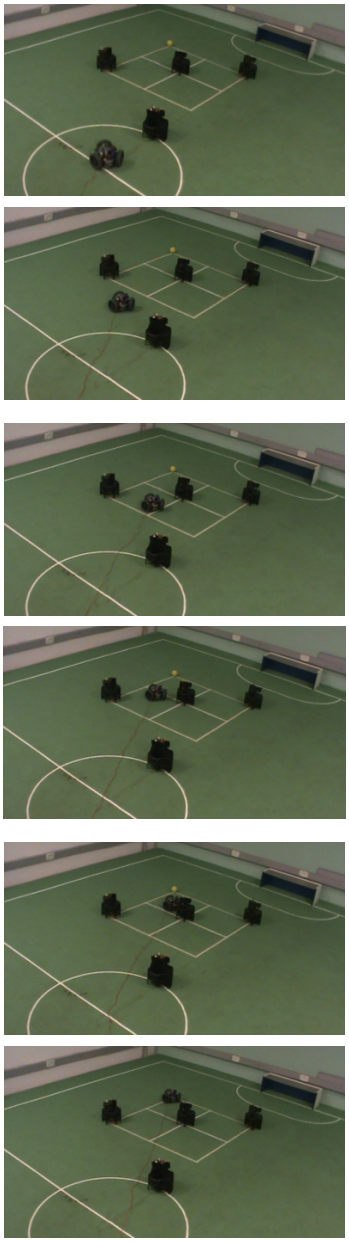

Fig. 8. Real Navigation using AxeBot

References

[1] L. Torrey, *Relational Transfer in Reinforcement Learning*, University of Wisconsin-Madison, Computer Sciences Department, 2009.

[2] A. Fatmi et al., *A Fuzzy Logic Based Navigation of a Mobile Robot*, In World Academy of Science, Engineering and Technology 22, 169–174, 2006.

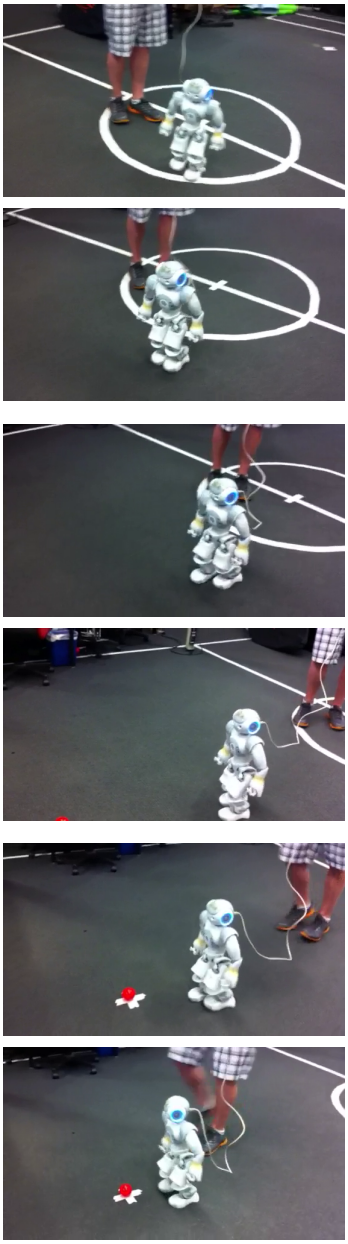

Fig. 9. Real Navigation using NaO

[3] R. Brooks, *Intelligence without representation*, In Artificial Intelligence 47, 139–159, 1991.

[4] M. Jaradat, M. Al-Rousan and L. Quadan, *Reinforcement based mobile robots navigation in dynamic environments*, In Robotics and Computer-Integrated Manufacturing 27, 135–149, 2011.

[5] T. Manikas, K. Ashenayi and R. L. Wainwright, *Genetic algorithms for autonomous robot navigation*, In IEEE Instrumentation and Measurement Magazine 10, 26–31, 2007.

[6] G. Bittencourt, *An embodied model for higher-level cognition*, In 29th Annual Meeting of the Cognitive Science Society (CogSci2007), 106–115, 2007.

[7] A.L. da Costa and G. Bittencourt, *From a concurrent architecture to a concurrent autonomous agents architecture*, Springer Lecture Notes in Artificial Inteligence LNAI 1856: Third International Workshop in RoboCup, 371–377, 1999.

[8] J. McCarthy and P. Hayes, *Some philosophical problems from the standpoint of artificial intelligence*, Machine Intelligence 4, 142–152, 1969.

[9] H. Ouarda, *A neural network based navigation for intelligent autonomous mobile robots*, In International Journal of Mathematical Models and Methods in Applied Sciences 4, 177–186, 2010.

[10] S. Pradhan, et al., *Navigation of multiple mobile robots using rule-based-neuro-fuzzy technique*, In International Journal of Computational Intelligence 3, 142–152, 2012.

[11] M. Shayestegan, *Mobile robot safe navigation in unknown environment*, In IEEE International Conference on Control System, Computing and Engineering, 44–49, 2012.

[12] L. Torrey, *Relational Transfer in Reinforcement Learning*, PhD thesis, University of Wisconsin-Madison, Computer Sciences Department, 2009.

[13] S. McGill et al., *Unified Humanoid Robotics Software Platform*, Proceedings of the 5th Workshop on Humanoid Soccer Robots Humanoids, 7–11, 2010.

.