| | |
|---|---|
| Project no.: | 027657 |
| Project full title: | Perception, Action & Cognition through Learning of Object-Action Complexes |
| Project Acronym: | PACO-PLUS |
| Deliverable no.: | D6.10 |

**Title of the deliverable:** **Scientific publication on a general decision making framework which integrates a learner, a planner and a human teacher to learn and execute complex tasks in human scenarios**

| | |
|---|---|
| Contractual Date of Delivery to the CEC: | 31. January 2010 |
| Actual Date of Delivery to the CEC | 12. February 2010 |
| Organisation name of lead contractor for this deliverable: | CSIC |
| Authors: | A. Agostini, F. Wörgötter, C Torras |
| Participants: | CSIC, BCCN |
| Work package contributing to the deliverable | WP6 |
| Nature: | R |
| Version | 1.0 |
| Total number of pages: | 21 |
| Start date of project: | 1st Feb. 2006 |
| Duration: | 52 months |

**Project co-funded by the European Commission within the Sixth Framework Programme (2002-2006)**

| **Dissemination Level** | | |
|---|---|---|
| **PU** | Public | **X** |
| **PP** | Restricted to other programme participants (including the Commission Services) | |
| **RE** | Restricted to a group specified by the consortium (including the Commission Services) | |
| **CO** | Confidential, only for members of the consortium (including the Commission Services) | |


**Abstract:**

This deliverable describes a working implementation of the OAC concept, the last version of the Rule Learning System (RSYS), and a description of the successful integration and synchronization of mechanisms within the three-level-architecture on ARMAR III. This last integration was done using a Decision Making Framework (DMF), which permits to close the learning-planning loop through perceiving and acting in the real world.

**Keyword list:** Learning OACs, Real Robots, Decision Making Framework, Integration of Mechanisms, Learning Action Rules, Closing Planning-Learning loop.

# Contents



## Executive Summary

The following three pages are the Executive Summary for this deliverable. They will summarize the technical report attached. Deviating from the original plan, we have not yet completed a scientific publication (in fact on paper has been rejected and needs now to be reworked and submitted elsewhere). Instead we are attaching the technical report here (on which this paper is based) as the report is more complete and describes all the relevant aspects.

## Implementation of the OAC concept on ARMAR

The OAC concept is instantiated through a structure denoted Probabilistic Cause-Effect Couple (pCEC). A pCEC clusters object-descriptions that, when present in a situation, afford an action execution. The confidence in the affordance is calculated probabilistically. The pCEC learning mechanisms was implemented on ARMAR III, as a module of the decision making framework (DMF) and was used for the generation or the refinement of rules and macro rules for planning during run-time.

## Implementation of the Decision Making Framework on ARMAR

A Decision Making Framework for closing the planning-learning loop, integrating, and synchronizing mechanisms within the three-level-architecture was successfully implemented on ARMAR III.

A brief description of the information flow in the DMF is given in the following: The signals obtained from the robot sensors are coded into a state representation suitable for symbolic reasoning. This is done mainly with mechanisms for object modelling and recognition. The current state and the task specifications are provided to a planner for decision making. The planner uses the rules or macro rules learned so far for plan generation. If a plan is found, the action of the first rule in the plan is decoded into suitable commands for the robot actuators. Then a new state representation is obtained and compared with the expected outcome of the executed rule. In case of outcome inconsistencies, the learning mechanisms in RSYS for automatic rule refinement are triggered.

In the test application the state representation was built from mechanisms for drinking cup recognition already implemented on ARMAR and the actual action execution was performed through grasping (and moving) those cups using visual servoing. The planner module of the DMF was instantiated in a cooperation with UEDIN from their PKS system.

### Integration of a human teacher in the planning-learning loop

The DMF also permits teacher intervention to support the planner in unrecognized situations. Teacher intervention is currently implemented for action instruction when no plan is found. Nevertheless, teacher intervention could be extended to any degree in the DMF, like, for instance, for supervision of perception and actions in conflicting situations.

## Contributions to PACO-PLUS

1. An **instantiation of the OAC** concept, and its implementation on ARMAR III.

2. A framework for the **integration of mechanisms within the three-level-architecture** to **close the planning-learning loop** iteratively.

3. A mechanism for **teacher intervention** when, for decision making, the robot needs help in conflicting situations.

4. Aspects 2 and 3 have been implemented on ARMAR III.

## Contributions beyond the state of the art

1. A mechanism for **on-line learning action rules and macro actions** for planning using **few experiences**.

2. A framework for **integrating different level mechanisms** for **closing the planning-learning loop** iteratively on a **real robot**.

Technical Report

IRI-TR-10-01

# Quick Learning of Cause-Effects Relevant for Robot Action

## IRI Technical Report

Alejandro Agostini
Florentin Wörgötter
Carme Torras



Institut de Robòtica i Informàtica Industrial

## Abstract

In this work we propose a new paradigm for the rapid learning of cause-effect relations relevant for task execution. Learning occurs automatically from action experiences by means of a novel constructive learning approach designed for applications where there is no previous knowledge of the task or world model, examples are provided on-line during run time, and the number of examples is small compared to the number of incoming experiences. These limitations pose obstacles for the existing constructive learning methods, where on-line learning is either not considered, a significant amount of prior knowledge has to be provided, or a large number of experiences or training streams are required. The system is implemented and evaluated in a humanoid robot platform using a decision-making framework that integrates a planner, the proposed learning mechanism, and a human teacher that supports the planner in the action selection. Results demonstrate the feasibility of the system for decision making in robotic applications.

**Institut de Robòtica i Informàtica Industrial (IRI)**
Consejo Superior de Investigaciones Científicas (CSIC)
Universitat Politècnica de Catalunya (UPC)
Llorens i Artigas 4-6, 08028, Barcelona, Spain
Tel (fax): +34 93 401 5750 (5751)
http://www.iri.upc.edu

**Corresponding author:**
Alejandro Agostini
tel: +34 93 401 5786
agostini@iri.upc.edu
http://www.iri.upc.edu/people/agostini

Copyright IRI, 2010

# 1 INTRODUCTION

In the last decades efforts have been made towards the development of a service robot capable of helping humans in human-like tasks. A requirement is a suitable description of the knowledge with a high-level of abstraction easily handled by humans, where objects and actions are described using logic statements. The need of grounding such knowledge for the robot to act in the real world implies the development of a suitable architecture that permits the integration of the different levels of abstraction involved, ranging from the mentioned highly abstract level to lower level mechanisms which are the last link in the chain for the robot embodiment.

The EU project PACO-PLUS [13] pursues the development of a cognitive system for service robots embedded in human environments. In this project it is claimed that the main building block for cognition is the Object-Action Complex (OAC). In brief, the OAC concept states that the world contains undistinguished ”things”, meaningless for the agent, which only become meaningful ”objects” through actions and tasks. For the application of OACs at different levels of abstraction, a cognitive architecture with three layers is proposed: the high, middle, and low level of abstraction layers. This architecture permits to articulate the different level mechanisms for a cognitive robot to be feasible. OACs are implemented at all levels, where the highest layer holds high-level OACs suitable for symbolic reasoning and human-robot interaction, and the lowest layer codes OACs for a direct performance in the real world.

In this work we propose an instantiation of the OAC concept in the highest level of abstraction using a structure denoted *probabilistic Cause-Effect Couple*, *pCEC* [2]. A *pCEC* describes not a single object but a category of objects using multiple sets of attributes. In a *pCEC* different combinations of attributes are generated and evaluated that, observed in a given object, would support an action affordance. The evaluation is done by associating to each combination of attributes a probability that estimates the chance of affording an action.

The estimation of the probabilities of action affordance can be considered as a classification problem the classification rule is the set of attributes, and classes are positive indicating an action affordance and negative in the converse case. We designed a novel probability estimation for a class of applications where experiences occur on-line, the amount of examples is small compared to the number of incoming experiences, and there is no previous knowledge of the world model or the task to be performed. In such applications, known classification metrics [11, 8], produce strongly biased estimations as demonstrated later in the paper. In general, they were designed for batch learning, when a large set of examples is provided in advance, and they do not take into account the uncertainties produced by the lack of experience. With respect to the generation of sets of attributes, we adopted the concept of perceptual categorization [7] which states that from all the object attributes that are relevant to afford an action, only a reduced number of them becomes relevant for a particular object instantiation. The learning of a relevant set of attributes is guided by this principle and is performed using a general to specific strategy, where multiple hypotheses of attributes combinations are maintained and evaluated.

The learning mechanisms proposed in this work are consistent with the human capability of learning cause-effect relations from experience, as enunciated by the Piaget’s theory of cognitive development [15], which claims that children gradually acquire knowledge of cause-effects relations by repeatedly executing processes and sequencing actions to reach goals.

The learned *pCEC*s can provide action rules for high-level reasoning and planning in the form of STRIPS-like planning operators. Any set of attributes in a *pCEC* could in principle be used to generate a rule but, to avoid the generation of action rules with little chance of action affordance, only sets of attributes with the highest probabilities are used for rule generation. Additionally to the described rule generation mechanism we developed another mechanism for the generation of macro rules, which contain not a single action but a sequence of actions playing the role of long-term cause-effects. These macro rules could significantly reduce the amount of deliberation as they might merge repetitive sequences of actions, or plans found with large computational cost. All the learned rules become immediately available for decision making and are refined and corrected constantly with the *pCEC*s improvements.

In order to implement the learning strategies and use *pCEC*s for decision making, the described mechanisms are embedded in a decision making framework where planning and learning occur iteratively, effectively closing the planning-learning loop. As we will see this decision making framework provides a suitable platform for the integration of the three levels of abstraction described in [13].

The rest of the paper is organized as follows. Next section introduces the *pCEC*, and in section III the learning mechanism of the *pCEC*s are described. Section IV explains how the *pCEC* are used to generate action rules for high-level reasoning. In section V experiments with a real robot are performed. The paper ends with some conclusions.

# 2 The Probabilistic Cause-Effect Couple (*pCEC*)

A *pCEC* maintains and generates hypotheses about which sets of attributes, when present in an object, would result in an action affordance. We say that an action is afforded if its execution leads to the expected changes in the world. These changes and the action uniquely characterize each *pCEC*, where the changes are coded using the attribute-values that change with the action, extracted from the prior and posterior states. The attribute-values modified with the action execution are used as a template for the generation of different sets of attributes to evaluate affordance, and the attributes after the change will describe the expected effects of the action.

## 2.1 Notation

In the following we refer to objects and states indiscriminately. We assume that an object is described with a set of $N$ attributes $d_i$, $i$=1,...,$N$. Each attribute can be instantiated in a set of discrete attribute-values $d_{ij}$, $j$=1,...,$|d_i|$. The representation of an observed object $s$ is constituted by a set of attribute-values $d_{ij}$, $s = \{d_{1j}, d_{2k}, ..., d_{Ng}\}$. We denote any subset of attribute-values as a subspace $ss$. In the same way, actions are represented symbolically, where each action references a low-level mechanism in charge of the actual action execution. The object descriptions before and after the execution of an action are named as the prior state $s_{prior}$, and the posterior state $s_{post}$, respectively. An affordance hypothesis $h_p$ is composed by any subsets of attributes that could be observed in prior states.

We describe a *pCEC* as a quadruple (1) composed by the affordance hypotheses $H_p$, the action $a$, the expected effects $ss_e$, and a set of experienced prior states $L_s$. In $L_s$ a state is labeled as a *positive* example if the action was afforded when the state was observed, and as a *negative* example in the converse case.

$$pCEC = \langle H_p, a, ss_e, L_s \rangle \tag{1}$$

# 3 Learning *pCEC*s

## 3.1 Birth of a *pCEC*

The birth of a *pCEC* takes place whenever the robot executes an action and observes changes never experienced before. Changes derived from the action execution are used as the substrate for the generation of the first affordance hypotheses in $H_p$, where all the possible specializations in one attribute-value are generated using as a template the set of attributes $ss_p^{new}$ involved in the change but before the action execution. The attribute-values modified with the action $ss_e^{new}$ will describe the expected effect of the *pCEC*.

$$ss_p^{new} = \{d_{ij} | d_{ij} \in s_{prior} \wedge d_{ij} \notin s_{post}\} \tag{2}$$

$$ss_e^{new} = \{d_{gk} | d_{gk} \notin s_{prior} \wedge d_{gk} \in s_{post}\} \tag{3}$$

### 3.2 Affordance Hypotheses. Evaluation.

The problem of evaluating the affordance of an action given a set of object attributes can be handled as a classification problem, where an observed object represents a training instance, the set of attributes is equivalent to a classification rule, and classes are *positive* for an action affordance, and *negative* in the converse case. Any classification method [11] could in principle be applied for this purpose but there are some important issues, inherent to the kind of applications we are dealing with, that should be considered. Conventional classification algorithms are designed for batch learning, where a significant amount of training instances are given in advance, while we are approaching an on-line kind of problem where instances are acquired incrementally from scratch. Additionally, they do not take into account the low confidence in the estimation caused by the lack of experience, and their classification metrics [8] produce biased estimations of action affordances when few examples have been observed.

We designed a novel evaluation metric that compensates the uncertainties originated by the lack of experience, giving unbiased predictions of the probabilities for a class. The uncertainties compensation is done by taking into account in the estimation not only the observed samples (objects), but also the states which have not been tried yet and are covered by the classification rule. The new metric is based on relative density of samples in a region covered by a classification rule, where the number of observed samples for a class is contrasted against the total number of states, both experienced and not experienced. Working with densities permits not only to take into account the accuracy in the estimation but also its confidence.

The designed evaluation criterion calculates the probability of the class $i$ for a classification rule $h$ in the following way,

$$P_h^i = \frac{1}{K}\left(1+(K-1)\frac{n_h^i}{n_h} - \sum_{\substack{j=1\\ j\neq i}}^{K}\frac{n_h^j}{n_h}\right) \tag{4}$$

where $n_h^i$ is the number of samples for class $i$ covered by $h$, $n_h$ is the total number of states covered by $h$, and $K$ is the number of classes.

For the case of a binary classification, and denoting the number of positive samples $p$, and negative samples $n$, equation (4) is enunciated for the positive class as,

$$P_h^+ = \frac{1}{2}\left(1+\frac{p}{n_h}-\frac{n}{n_h}\right) \tag{5}$$

It is straigthforward to see that the probability of a negative is,

$$P_h^- = \frac{1}{2}\left(1+\frac{n}{n_h}-\frac{p}{n_h}\right) = 1-P_h^+ \tag{6}$$

With these formulas, a high $P_h^+$ is a confident indicator of the chance of obtaining a positive as equation (5) assigns to unexplored states the same chance to result in a positive or a negative, and compensates the estimation given by the positive examples with the negative experiences. For instance, statistics fed only a few times with successful examples would result in a $P_h^+$ a little higher than 0.5, while the other evaluation criteria [8] would indicate a high chance of obtaining a success even with a few examples.

To illustrate the benefits of using the proposed density metric in cases where few examples are observed, we compare its outcomes for different situations with the most well known metrics for predictive performance [8]. The metrics used in the comparison are presented in table 1 for the case of binary classification, where $N$ accounts for the total number of negative examples observed and $P$ for the positive ones.

The table shows the formula for Entropy, m-estimate, Laplace, and Weighted Relative Accuracy (WRA). The Entropy metric estimates the uniformity of the classes in the region covered by the classification rule, and it is used by many learning methods, like the information gain in decision trees [11],

Table 1: Hypothesis Evaluation Metrics

| | |
|---|---|
| Entropy | $P_h = -\left(\frac{p}{p+n}\log_2\frac{p}{p+n} + \frac{n}{p+n}\log_2\frac{n}{p+n}\right)$ |
| M-estimate | $P_h = \frac{p+m\frac{P}{N+P}}{n+p+m}$ |
| Laplace | $P_h = \frac{p+1}{p+n+2}$ |
| WRA | $P_h = \frac{p+n}{P+N}\left(\frac{p}{p+n} - \frac{P}{P+N}\right)$ |

Table 2: Performance Comparison for two hypotheses ($h_1$, $h_2$) given a few experiences.

| Hypo | $p$ | $n$ | $n_h$ | $P$ | $N$ | Ent | m-est | Lapl | WRA | Dens |
|---|---|---|---|---|---|---|---|---|---|---|
| $h_1$ | 3 | 0 | 200 | 3 | 1 | 0,00 | 0,81 | 0,80 | 0,19 | 0,51 |
| $h_2$ | 3 | 1 | 50 | 3 | 1 | -0,81 | 0,75 | 0,67 | 0,00 | 0,52 |

or the first version of the CN2 algorithm [6]. We take the negative value of the entropy so to have lower values for worst rules. The *m*-estimate provides the probability of a class for an example covered by a classification rule. The probability is biased by the prior probability $p(c)$ for the class, weighted by a tunable parameter *m*, which regulates the influence of the prior probability in the rule inference: higher values of *m* produce larger influence of the prior probability of the class, while lower values diminish the influence of $p(c)$ in the rule inference. In the special case $m = 0$, the inference of a rule is given by the traditional relative frequency. The prior probability $p(c)$ may be calculated from the total amount of observed samples using the relative frequency formula, as shown in the table, or may be calculated as $1/K$, where $K$ is the number of classes. Such is the case of the Laplace estimation, which is a special case of the *m*-estimate, where the parameter *m* is set to $K$. Laplace formula has been used, among others, by the improved version of the CN2 method [5]. Finally, WRA was one of the most recently developed metric, and it provides a performance evaluation that compensates the accuracy of a classification rule with the accuracy of the universal rule, which covers the whole set of examples, to take into account the actual improvements in the estimation. This relative accuracy is weighted by the probability that an example is covered by the classification rule to take into account the generality of the rule in the predictive performance. Note that all these metrics take into account only the observed examples, a none of them considers the uncertainties in the estimations produced by the lack of experiences.

In table 2 the metrics for a couple of classification rules, $h_1$ and $h_2$, are compared when only a few examples were experienced so far. The total number of possible samples to be experienced (i.e. not yet tried objects when talking about affordances) for $h_1$ is 200, and for $h_2$ is 50. In this case all the metrics selects rule $h_1$ as the best rule for the prediction, while the proposed density metric selects rule $h_2$ as it has more chances of observing positive examples in the future, even having one negative experience. Note the high probability assigned by Laplace and *m*-estimate to rule $h_1$. In contrast, the density metric produces probabilities slightly higher than 0.5.

There are some points to remark. First, the criterion $WRA$ takes into account the total number of positive and negative samples, $P$ and $N$, to compensate the estimation given by the rule. As the total number of samples increases, the bias in the estimations of $WRA$ for $h_1$ decreases, and the hypothesis selection is compensated towards $h_2$ (table 3). Nevertheless, its estimation still produces $h_1$ selection even when the total number of samples increases to $\infty$.

Finally, in order to illustrate the behavior of our approach under a large number of examples, we perform the calculations using arbitrary large experiences (table 4). In this case the proposed density criterion produces the same rule selection as the other criteria.

Table 3: Performance Comparison for two hypotheses ($h_1$, $h_2$) when the total number of experiences increases.

| Hypo | $p$ | $n$ | $n_h$ | $P$ | $N$ | Ent | m-est | Lapl | WRA | Dens |
|---|---|---|---|---|---|---|---|---|---|---|
| $h_1$ | 3 | 0 | 200 | 3E4 | 1E4 | 0,00 | 0,81 | 0,80 | 2E-5 | 0,51 |
| $h_2$ | 3 | 1 | 50 | 3E4 | 1E4 | -0,81 | 0,75 | 0,67 | 0 | 0,52 |

Table 4: Performance Comparison for two hypotheses ($h_1$, $h_2$) after arbitrarily large number of experiences.

| Hypo | $p$ | $n$ | $n_h$ | $P$ | $N$ | Ent | m-est | Lapl | WRA | Dens |
|---|---|---|---|---|---|---|---|---|---|---|
| $h_1$ | 100 | 10 | 200 | 1E3 | 1E3 | 0,44 | 0,88 | 0,90 | 0,02 | 0,73 |
| $h_2$ | 20 | 10 | 50 | 1E3 | 1E3 | -0,92 | 0,63 | 0,66 | 0,00 | 0,60 |

We have illustrated that the new proposed evaluation criterion avoids giving biased estimations mainly at the beginning of the on-line learning process, where few examples are provided relative to the total possible number of experiences.

### 3.3 Affordance hypotheses. Generation.

As mentioned before, the generation of hypotheses is based on the concept of perceptual categorization [7] which states that only a reduced number of attributes would result relevant for the estimation of the probability of affodance of a particular observed object. Therefore, the learning of a relevant set of attributes is performed using a general to specific strategy, where multiple hypothesis of attributes combinations are maintained and evaluated.

Generation is triggered when an action is tried but not afforded, fact denoted as a *surprise*. An action is not afforded when the consistency check between the predicted and the obtained effects fails, i.e. $ss_e \notin s_{post}$.

In the case of a surprise a best-first strategy is used for the generation of new hypotheses: all the specializations in one attribute-value of the "best" hypothesis are generated. The best hypothesis, which we call the *winner* hypothesis $h_w$, is that with the highest chance of affordance, and, in case of a draw among many hypotheses, with fewer attributes.

The inconsistency between the obtained and the expected outcome could be produced by many factors. Reasons can be found in a bad action model, random unpredictable changes, random contingencies that prevent the outcome to occur, or incomplete or wrong preconditions consideration. In this work we assume that actions are correctly modeled and that the world is deterministic. Under these circumstances, failures in the outcomes are only produced by an incomplete or wrong affordance hypothesis.

Finally, after the generation of a hypothesis, the statistics associated are initialized according to the stored examples in $L_s$.

## 4 Using *pCEC* for High-Level Decision Making

### 4.1 Instantiating Planning Operators from pCECs

In order to use *pCEC*s for high-level reasoning, STRIPS-like rules [10] are generated from the *pCEC*s using as rule preconditions the winner hypothesis. We describe a STRIPS-like rule $R_i$ as a triple composed by a precondition part $ss_e^p (= h_w)$, an action part $A_i$, and the effects $ss_e^i$, where the action and the effects are those form the *pCEC*.

$$R_i = \{A_i, ss_p^i, ss_e^i\} \tag{7}$$

For simplicity, we work under the closed world assumption, so all the conditions that are not true for a detector are assumed to be false. This allows us to save the deletions in the effect part of the rule.

We would like to remark that there are other approaches to learn STRIPS-like rule, but the limitations aforementioned about the kind of problems we are facing pose unsolvable obstacles to them, where on-line learning is either not considered [12, 18], a significant amount of prior knowledge has to be provided [9], or a large number of experiences or training streams for the learning of behaviors are required [17, 18, 4, 16, 12].

## 4.2 Refining Planning Operators from Affordance Hypotheses

When using *pCEC*s for decision making, actions executions are guided by the planner (see below). Whenever an action is not afforded, new affordance hypotheses are generated using the mechanisms described in section 3.3 and the corresponding action rule is refined by replacing its precondition part with the new winner hypothesis.

## 4.3 Macro Rules

Rule $R_i$ may reference a single action $A_i = a_k$, when derived from a *pCEC*, or a sequence of actions $A_i = \{a_j, a_k, ..., a_m\}$, when generated as a macro rule. For the learning of macro rules, we developed a technique of condition propagation [1] that permits to find the necessary preconditions that will produce the cumulative changes in the world obtained after a sequence of action executions. For a more detailed explanation about macro rule generation please refer to [1].

It is important to mention that, when a single action rule is refined, all the macro rules that involve it are refined too using the condition propagation method.

## 4.4 Decision-Making Framework

Figure 1 depicts a flow diagram for the general framework of decision making in which the learning system is embedded. The framework is composed of a planner, which build plans to accomplish tasks with the rules learned so far, the rule learning system, which provides and refines rules for the planner, and a teacher interface, that permits teacher instruction about actions to perform when no plan is found.

Learning *pCEC*s is supported by experimenting actions in the environment. The action selection may be guided using different strategies, which can be random (exploration) or deterministic (exploitation). In order to avoid large unnecessary exploration of actions, action selection is done in a supervised way to speed the learning for task execution. Actions to be executed are provided by a logic-based planner, and - if the planner fails because of incomplete knowledge - by a teacher through action instructions. In this way, we take benefit of human natural knowledge about the expected cause-effects in currently observed situations: it is very simple for a human to know which action to perform in a situation given a plain task, but it could be much more complicated to explain a priori all the sequences of actions that should take place in all possible situations. Hence, teacher instructions simply consist of a single action to be performed in the current situation according to the task in progress. In this work, we let the teacher control the macro rule generation by the instructions given. The teacher could instruct a sequence of actions, one for each observed situation after the generation of a rule, whenever he considers convenient to store the sequence into a macro rule. An action instruction produces a new *pCEC* which produces in turn action rules to fill the gap of knowledge.

Planning and learning occur iteratively during run-time. The action to be executed is dictated by the planner using the first action of the generated plan, and if no plan is found due to the lack of rules, by a human teacher. The planner searches for plans to accomplish the goal from the current situation, and produces one of three possible outcomes. If the goal requested is already reached, the system halts until the next goal specification. In the case a plan is found, the planner yields the first action of the plan. This

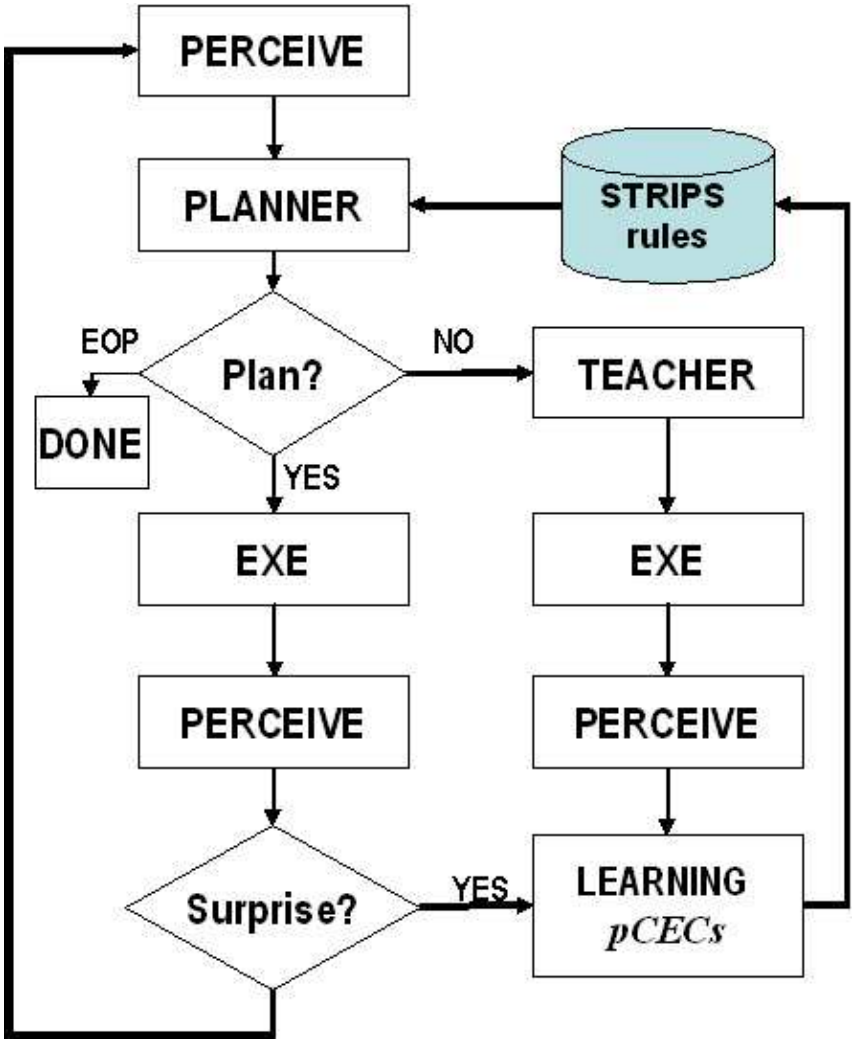


Figure 1: Schema of the decision-making framework that integrates the robot, the learning mechanism, the planner and a human teacher through suitable interfaces.

action is executed and the outcome is evaluated. If the outcome is not consistent with the expected one, fact that we denote as *surprise*, the rule refinement mechanism is triggered to solve the inconsistencies.

# 5 Using *pCEC* for Task Execution on Humanoid Robot

The decision making framework was implemented on the humanoid robot platform ARMAR III [3] so to assess the validity of the approach. The example application consists in arranging cups on a sideboard to avoid collisions when moving a cup from one position to another. Given a task specification consisting of a cup to be moved, denoted as the target cup, and its desired destination the robot should learn to move the cup to the specified position without colliding with other cups. For this, it should move cups that could interfere with the movements in an ordered way.

It is important to remark that the aim of this example is to illustrate the system mechanisms and their reliability on a real complex robot, and not to solve such a simple task, which can be indeed solved by many other known strategies. As the aim of the implementation is to evaluate the *pCEC*s learning and their use for high-level decision making, the perceptions and actions mechanisms of the robot were simplified as much as possible to permit a rapid and clean implementation. Note that the set of perceptions and actions of the robot can be of any nature as far as they provide symbolic references to lower-level mechanisms, no matter how sophisticated or simple these mechanisms are. The example presented is rather simple but settles the foundation for the learning of more complex tasks.

## 5.1 Perceptions

In the example task, for the representation of the position of the cups, we adopt the simple strategy of partitioning the sideboard into cells using a grid world. Each cup is considered to lie inside a cell. This requires the robot to perform precise movements of the cups so to place them in positions close to the center of the cells and avoid false detections. Formally, for each cell $i$, an attribute $d_i$ is considered, with $i$=1,...,$N$. $N$ is the total number of cells and depends on the partition made. An attribute $d_i$ could be instantiated in one of three possible attribute-values,

- $d_{i1} = e(i)$, true if cell $i$ is empty.
- $d_{i2} = o(i)$, true if a cup is inside cell $i$.
- $d_{i3} = to(i)$, true if the target cup is placed on cell $i$.

The target cup is identified with a color provided by the user (green in the example) and referenced as $to$. In rule activation, $o(i)$ and $to(i)$ are considered equivalent.

## 5.2 Actions

In the application, actions are performed through pick and place with grasping. The robot is limited to perform simple straight movements of the cups in the horizontal or vertical direction. Then, an action is defined by the direction and the number of cells of the displacement. In the figures, actions are described using three digits, the first accounts for the index of the cell where the object is placed, the second is the direction of the movement, and the third one is the number of cells in that direction. The direction of the movements can adopt one of four possible movements with respect to the robot: $U$ for the forward movement, $D$ for moving an object backward, $L$ for moving to the left, and $R$ for movements to the right.

Figure 2 presents the problem definition graphically.

## 5.3 The Planner

The planner used for deliberation with the generated action rules is the PKS logic-based planner [14] which uses STRIPS-like rules for plan generation.

## 5.4 Experiments

Results of four experiments are presented for the task of moving the green cup from cell 5 to cell 7. In the first experiment (figure 3) the robot faces a situation with no blocking objects and initiates with no rules in the data base, being not possible to find a plan. This produces an action instruction from the teacher, which instructs the action *5R2* ("move object in cell 5 two positions to the right"). The robot executes the action and generates $pCEC_1$ and the rule $R_1$ from the observed changes. The generated *pCEC* is depicted in figure 4. The affordance hypothesis with highest chance of a positive and fewer attributes is used to generate the precondition part $ss_p^1$ of rule $R_1$. Figure 4 also shows the probabilities associated to each affordance hypothesis using a bar representation, and how they are updated with the first positive sample.

In experiment 2 (figure 5) the robot is asked again to move the green cup two positions to the right, but this time a blocking cup is placed in the trajectory. As the only rule generated so far, $R_1$, does not contemplate the middle cell in $ss_p$, the robot uses it to cope with the goal. In this case, instead of executing action 5*R*2, we bypass it to avoid collision. Nevertheless, ARMAR believes that it was indeed executed and evaluates the obtained outcome, which produces a surprise as it is not consistent with the observation. Then, affordance hypotheses generation is triggered together with the rule $R_1$ refinement. Figure 6 graphically depicts the learning mechanisms in $pCEC_1$. First, all the specializations in one condition of the failed affordance hypothesis are generated, and their statistics are initialized according to the stored examples. Then, the affordance hypothesis that most likely affords the action, and with fewer attributes, is used for $ss_p^1$ refinement. After rule correction, the robot faces again the same initial situation but this time no rule is applicable (because of the added attribute-value $e(6)$). The action instruction mechanism is triggered and the teacher instructs the action needed to remove the blocking object, "*6U1*". Finally, the freed path permits to reach the goal with rule $R_1$.

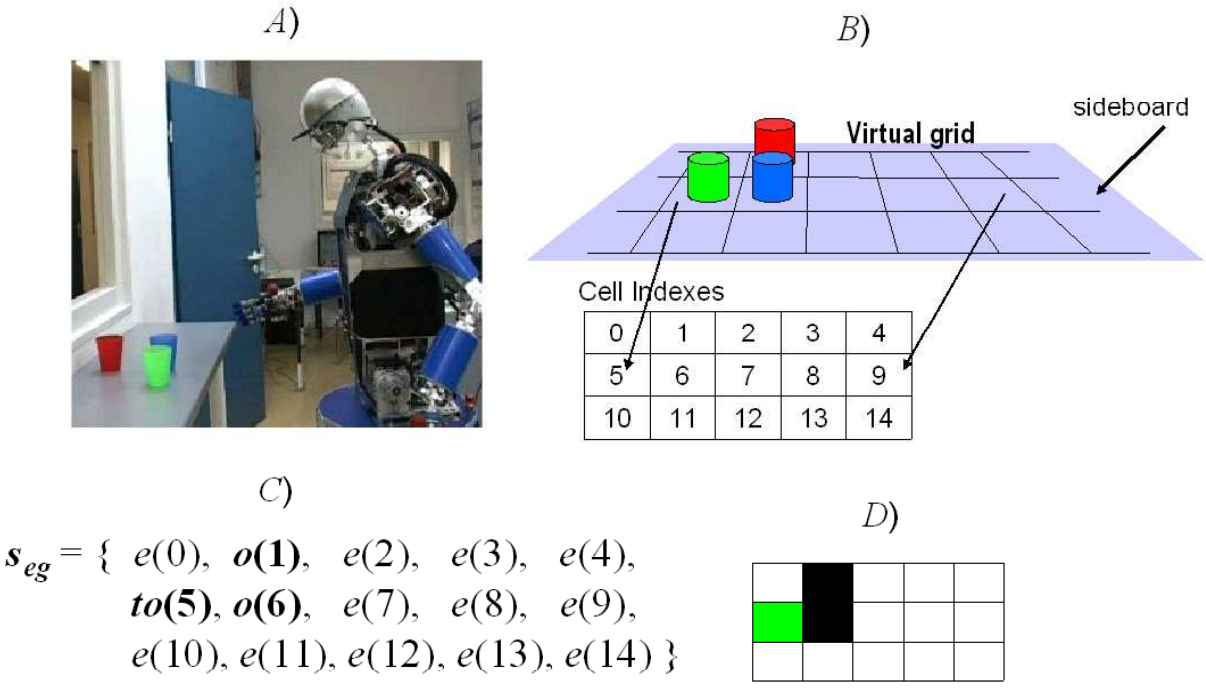


Figure 2: Problem definition for the task of arranging cups. A) Real setting. B) Graphical representation of the robot perspective. C) Example state $s_eg$. D) graphical representation of the state, where the cell colored in green corresponds to the target cup, and white and black cells refer to empty or occupied cells, respectively.

The third experiment (figure 7) consists of a more complicated situation with many cups in the scenario. The milestone of this experiment is the generation of a macro rule. This experiment is not consecutive to the previous ones and one more rule has been generated in between. Figure 8 shows the set of rules learned so far when experiment 3 was performed. The initial situation prevents the planner to find a plan and an action instruction is needed at the first stage of the process. The teacher instructs, first, the action to move the cup in cell 6 one position to the right, "*6R1*", which leads to a *pCEC* and rule generation, and afterward the action "*7U1*". Every instructed action produces the corresponding *pCEC* and rule generation. After the second instruction, the teacher quits instructing actions, and, as two actions were instructed, the macro rule $R_6$ is generated. $R_6$ is composed of $R_4$ and $R_5$, and its preconditions and effects are obtained through the conditions propagation method [1]. From the resulting situation the goal can be now reached using $R_1$.

Finally, in experiment 4 (figure 9) the rules learned in experiment 3 are evaluated using the same initial situation as experiment 3. The planner is now able to find a plan consisting in the chaining of rule $R_6$ and $R_1$. The execution of the macro rule is performed through the execution of each rule it contains, and the surprise evaluation is carried out at each step. In this experiment, no surprises arise and the task is completed successfully.

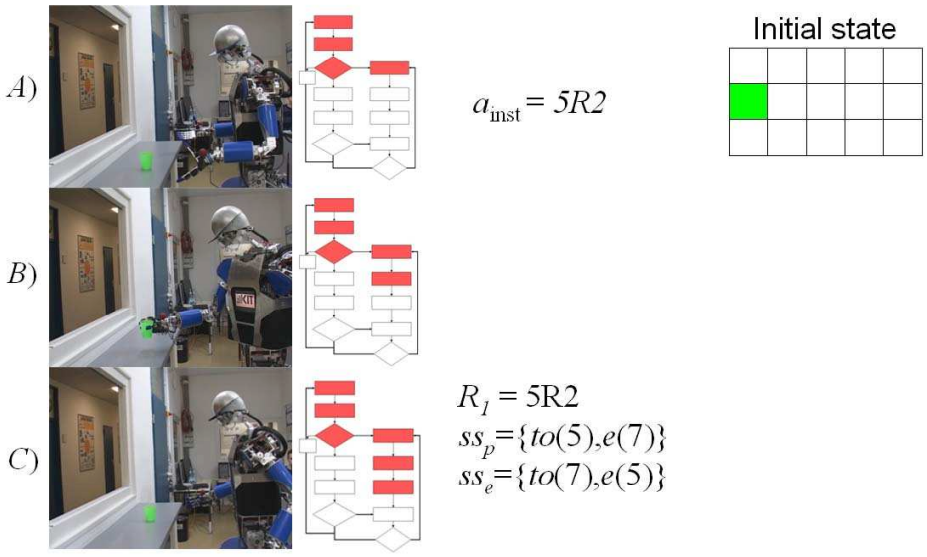


Figure 3: Experiment 1. No blocking object.

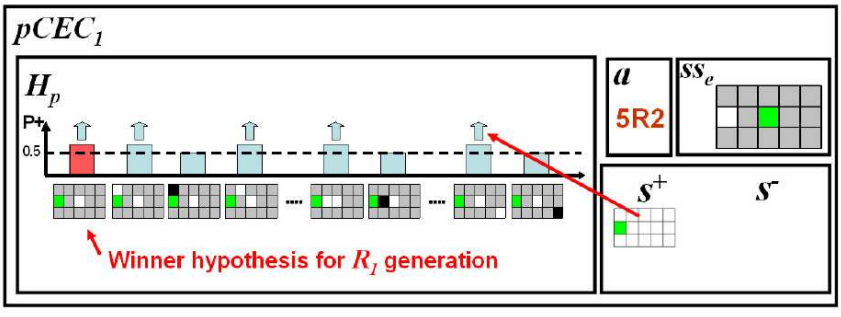


Figure 4: Generated $pCEC_1$. The affordance hypotheses are represented graphically. Gray cells indicate a "don't care" whether the cells are occupied or not.

# 6 Conclusions

This work presents a system to rapidly acquire cause-effect rules for decision making using a new paradigm for representing and learning object categories. The method is based on the concept of OAC which states that objects are defined in accordance to their use, where objects and actions are inseparably intertwined. Learning of cause-effects occurs automatically from changes observed after action executions using a novel learning approach that permits the robot to automatically learn relevant set of attributes to afford actions from few examples. Additionally, we implemented a decision making framework that permit to rapidly acquire and use learned cause-effects for decision making, where learning occurs iteratively with planning steps, closing effectively the planning-learning loop. The inclusion of

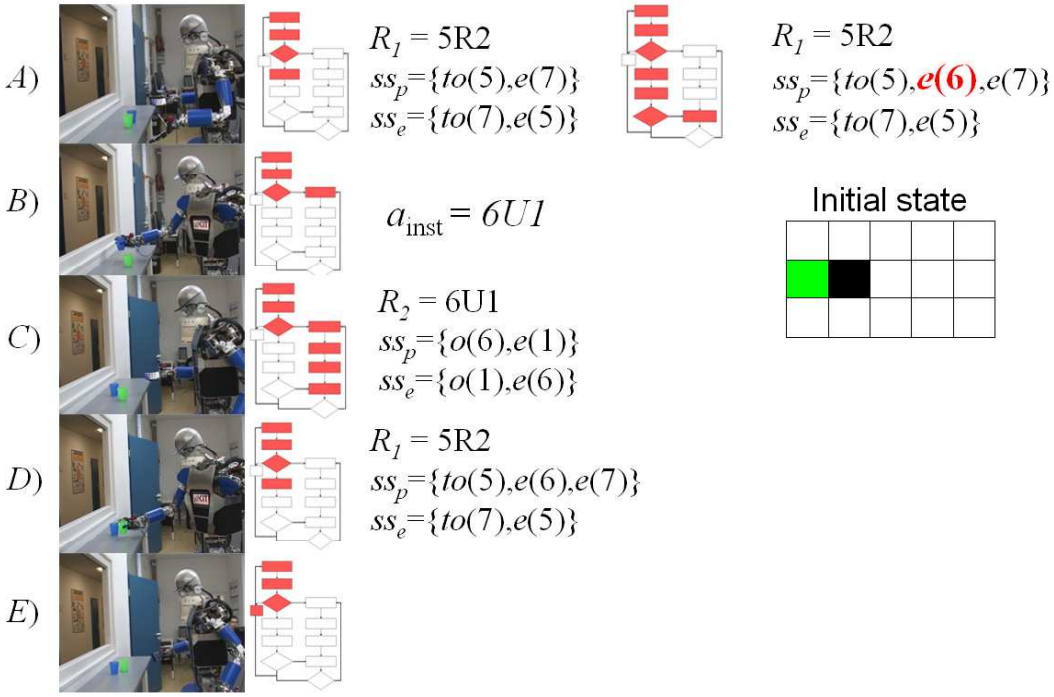


Figure 5: Experiment 2. One blocking object. The figure shows a surprise with the following rule refinement using the winner hypothesis.

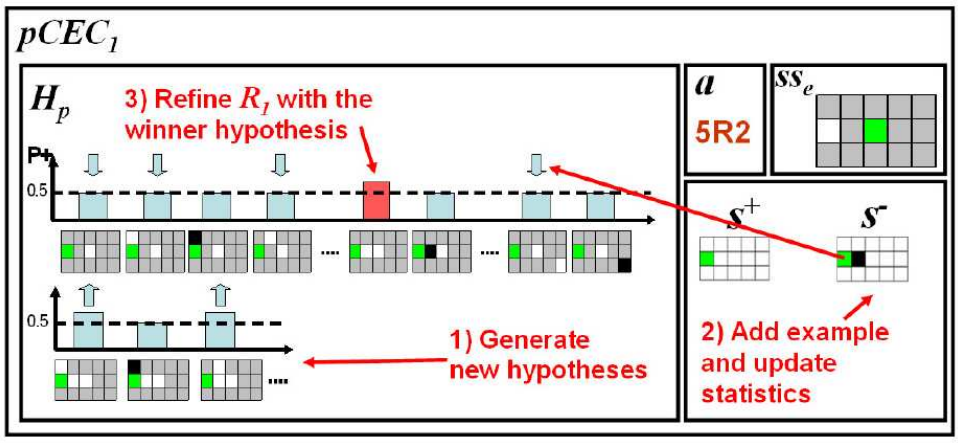


Figure 6: Learning processes in $pCEC_1$ under a surprise. 1) Generation of all the specialization of the best hypothesis. 2) Updating of the statistics using the negative example. 3) Rule $R_1$ refinement using the winner hypothesis.

a human teacher in the planning-learning loop to instruct single actions when the planner fails to find a

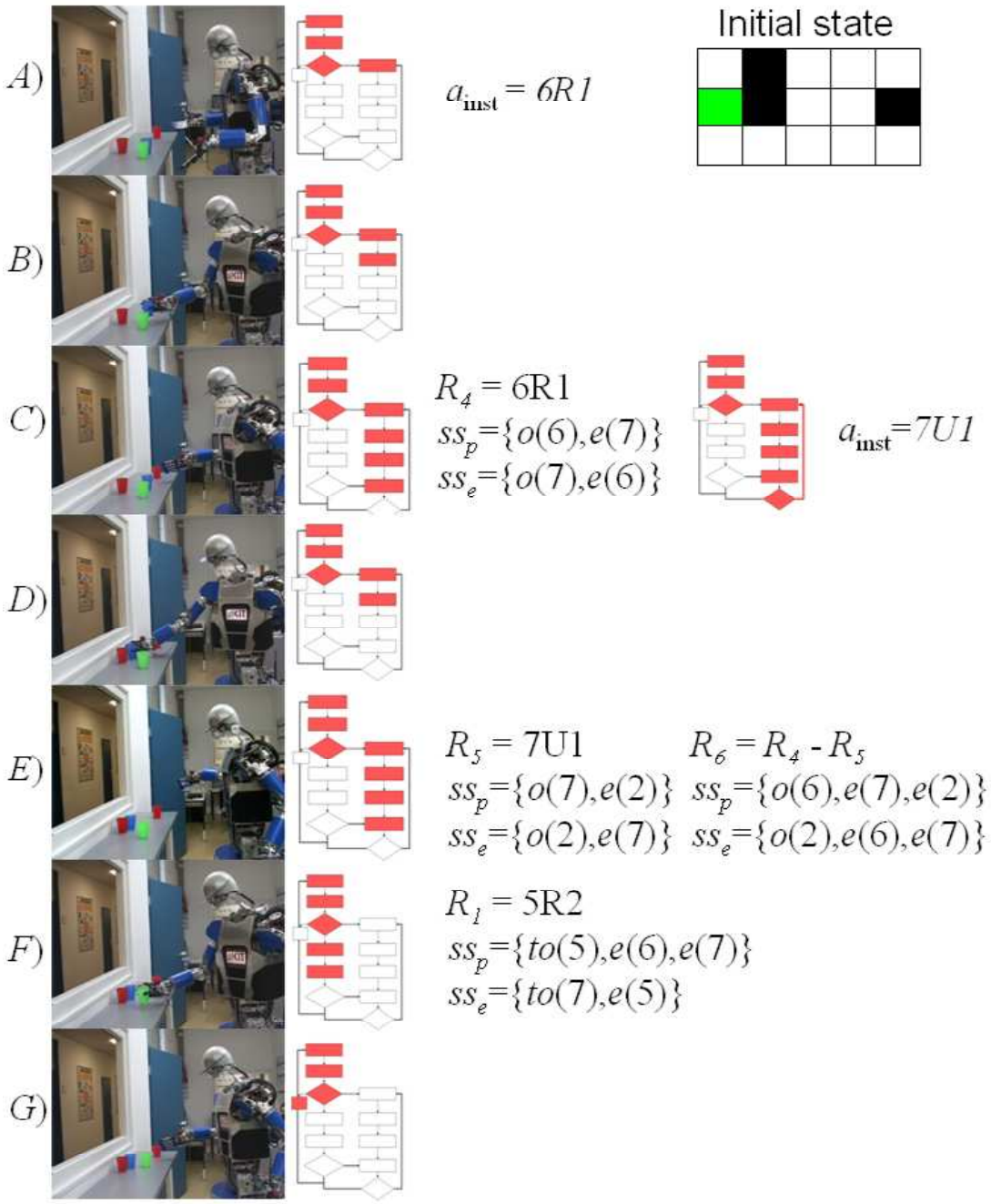


Figure 7: Experiment 3. Many blocking objects. More rules generation are shown, including a macro rule of two steps.

$R_1 = 5R2$
$ss_p = \{to(5), e(6), e(7)\}$
$ss_e = \{to(7), e(5)\}$

$R_2 = 6U1$
$ss_p = \{o(6), e(1)\}$
$ss_e = \{o(1), e(6)\}$

$R_3 = 6R3$
$ss_p = \{o(6), e(9)\}$
$ss_e = \{o(9), e(6)\}$

Figure 8: Initial set of rules in experiment 3.

plan permits to significantly accelerate the learning of task-relevant cause-effects.

We implemented and tested the system in a real complex platform of the humanoid robot ARMAR III. The results elucidate the reliability of the system in a real scenario to learn and perform tasks in human environments.

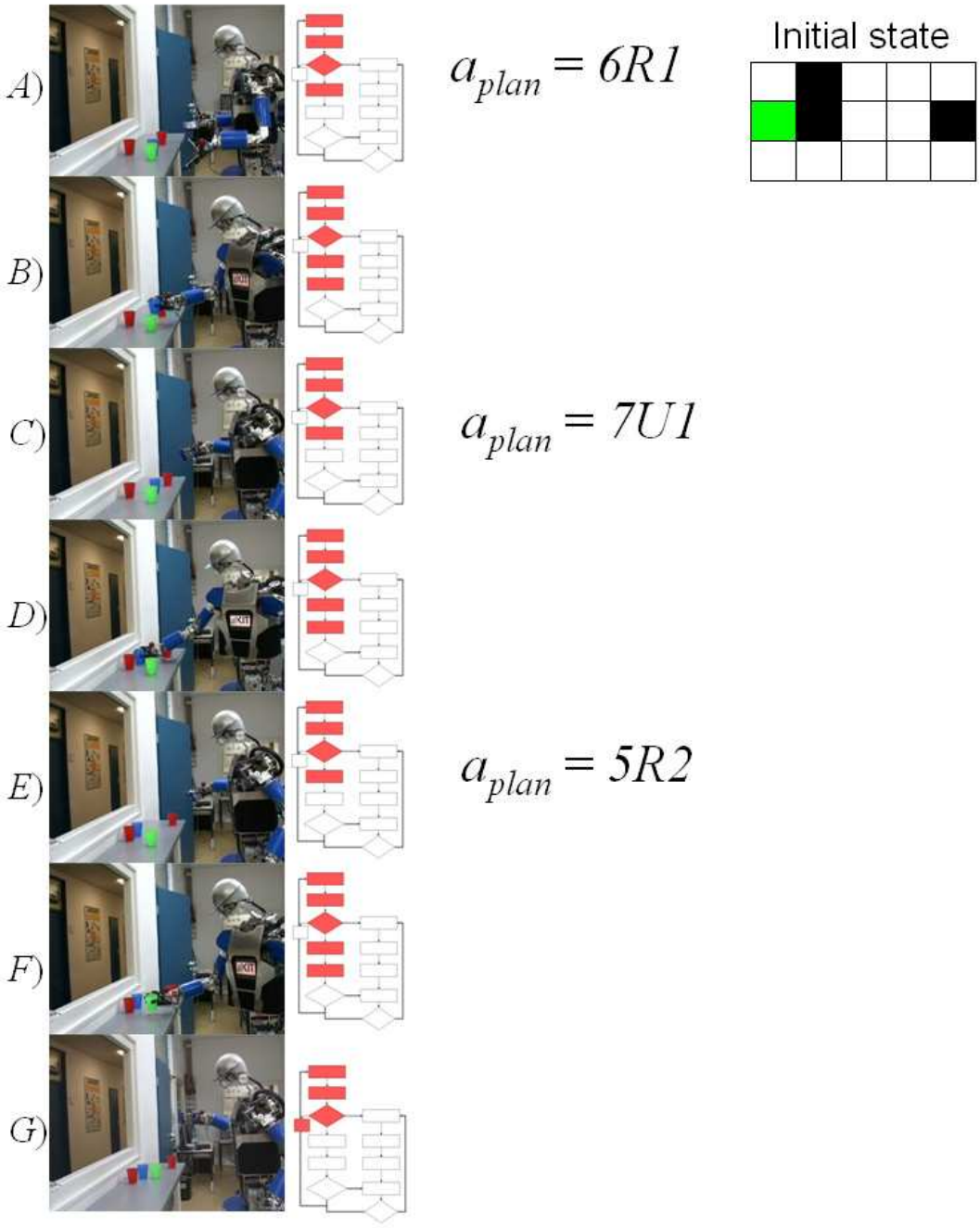


Figure 9: Experiment 4. Evaluation of rules learned in experiment 3.

## References

[1] A. Agostini, E. Celaya, C. Torras, and F. Wörgötter. Action Rule Induction from Cause-Effect Pairs Learned Through Robot-Teacher Interaction. In *In Proc. of the International Conference on Cognitive Systems, CogSys 2008. (Karlsruhe, Germany)*, pages 213–218, 2008.

[2] A. Agostini, F. Wörgötter, E. Celaya, and C. Torras. On-Line Learning of Macro Planning Operators using Probabilistic Estimations of Cause-Effects. Technical report, IRI-TR 05/2008. Institut de Robòtica i Informàtica Industrial. UPC-CSIC. (Barcelona, Spain), 2008.

[3] T. Asfour, P. Azad, N. Vahrenkamp, K. Regenstein, A. Bierbaum, K. Welke, J. Schroeder, and R. Dillmann. Toward humanoid manipulation in human-centred environments. *Robotics and Autonomous Systems*, 56(1):54 – 65, 2008.

[4] S. Benson. Inductive learning of reactive action models. In *Proceedings of the Twelfth International Conference on Machine Learning*, pages 47–54. Morgan Kaufmann, 1995.

[5] P. Clark and R. Boswell. Rule induction with CN2: Some recent improvements. In *Proceedings of the Fifth European Working Session on Learning*, pages 151–163. Springer, 1991.

[6] P. Clark and T. Niblett. The CN2 induction algorithm. *Machine learning*, 3(4):261–283, 1989.

[7] G. Edelman and N. Darwinism. The theory of neuronal group selection. *Neural Darwinism*.

[8] J. Furnkranz and P.A. Flach. An analysis of rule evaluation metrics. In *Proceedings of the Twentieth International Conference on Machine Learning. ICML03.*, volume 20, pages 202–209, 2003.

[9] Y. Gil. Learning by experimentation: incremental refinement of incomplete planning domains. *Proceedings of the Eleventh International Conference on Machine Learning*, 1994.

[10] S. M. LaValle. *Planning Algorithms*. Cambridge University Press, Cambridge, U.K., 2006. Available at http://planning.cs.uiuc.edu/.

[11] T. Mitchel. *Machine Learning*. WCB McGraw Hill, 1997.

[12] T. Oates and P. Cohen. Learning planning operators with conditional and probabilistic effects. In *In Proceedings of the AAAI Spring Symposium on Planning with Incomplete Information for Robot Problems*, pages 86–94. AAAI, 1996.

[13] PACO-PLUS. PACO-PLUS: perception, action and cognition through learning of object-action complexes. http://www.paco-plus.org/, 2006-2010.

[14] R. Petrick and F. Bacchus. A knowledge-based approach to planning with incomplete information and sensing. In Malik Ghallab, Joachim Hertzberg, and Paolo Traverso, editors, *Proceedings of the Sixth International Conference on Artificial Intelligence Planning and Scheduling (AIPS-2002)*, pages 212–221. AAAI Press, 2002.

[15] J. Piaget. *The origins of intelligence in children (trans. M. Cook)*. New York: International Universities Press, 1952.

[16] W. Shen. Rule creation and rule learning through environmental exploration. In *In Proceedings IJCAI-89*, pages 675–680. Morgan Kaufmann, 1989.

[17] T.J. Walsh and M.L. Littman. Efficient learning of action schemas and web-service descriptions. In *Proceedings of the Twenty-Third AAAI Conference on Artificial Intelligence*, pages 714–719, 2008.

[18] X. Wang. Planning while learning operators. In *AIPS*, pages 229–236, 1996.

## Acknowledgements

I would like to thanks Dr. Tamim Asfour and Prof. Rüdiger Dillmann from the Karlsruhe Institute of Technology for all the support given, and for letting me implement the RSYS on the ARMAR platform. I am also very grateful with their team, in special with Stefan Ulbrich, Nils Adermann, David Gonzalez, and Manfred Kroehnert, who made an enormous effort to make the implementation possible. This work is funded by the EU PACO-PLUS project FP6-2004-IST-4-27657.

## IRI reports

This report is in the series of IRI technical reports.
All IRI technical reports are available for download at the IRI website
`http://www.iri.upc.edu`.