roject no.:	027657
Project full title:	Perception, Action & Cognition through Learning of Object-Action Complexes
Project Acronym:	PACO-PLUS
Deliverable no.:	D1.2.3

Title of the deliverable: Technical report on the final integration in PACO-PLUS, the cognitive control architecture and the use of OACs in the architecture.

Contractual Date of Delivery to the CEC:	31. May 2010
Actual Date of Delivery to the CEC	12. July 2010
Organisation name of lead contractor for this deliverable:	KIT/SDU
Authors:	T. Asfour, R. Petrick
Participants:	All
Work package contributing to the deliverable	WP1
Nature:	R & D
Version	1.0
Total number of pages:	8
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:

The deliverable describes the final integration of different components developed in the project to support the final demonstration on the project's achievements in terms of implementation of the architecture and demonstration of the acquisition, learning and interplay of OACs within the cognitive control architecture. The deliverables can be seen as a revised version of D1.2.2 and D4.2.3 which were submitted in the last reporting period. The demonstration of the integrated components has been reported in D8.3.

Keyword list: Integration, architecture

Table of Contents

1. Executive Summary

The integration efforts in PACO-PLUS have been continued in the last year of the project. The software architecture of ARMAR has been constantly extended to cope with the requirements for the implementation of the developed PACO-PLUS methods and concepts.

In the last year we reported on the integration of various capabilities in the control architecture of ARMAR and provided demonstrations of these capabilities on ARMAR (cf. D1.2.2, D8.2.3, D8.2.4). In the last period the implementation of both the developed capabilities as well as the control flow algorithms were tested and made robust to ensure smooth interplay between the implemented OACs, thus providing solid basis for the implementation of the integrated final demonstration.

The capabilities (see D1.2.2 and D8.3) consists of

- Strategies for grasping known and unknown objects through combining various methods and techniques.
- Learning of pushing actions for grasping
- Learning early object-action complexes by imitation and coaching
- Active vision methods for various visual tasks such as visual search, active observation of human actions and on-line imitation of human actions
- Grasp recognition and mapping
- Haptic exploration with a five-fingered hand
- Symbolic Planning for the kitchen scenario

In addition, several components have been successfully implemented and demonstrated on other robot platforms in the project. Among those are a) pouring and filling on HOAP b) learning of grasp densities on the SDU platforms, c) 3D vision and action learning on CBi at ATR, Japan.

2. Extension of the ARMAR API

The work concentrated on the final integration of both low-level as well as high level components. Software interfaces have been extended and refined for the final integration within the 3-level cognitive architecture. Further, integration of numerous components from other work packages on the humanoid robot ARMAR-III has been finalized. Additional details about these components have been reported in D1.2.2 and D8.3.

For the final integration on ARMAR, several extensions of ARMAR’s API (see Deliverable 1.1.1) have been performed to provide a convenient access to the robot’s hardware and the implementation of more complex tasks such as the final demonstration. The implemented capabilities are

- `SearchForObjectSkill`
- `VisualGraspSkill`
- `TwoArmVisualGraspSkill`
- `VisualServoingSkill`
- `TwoArmVisualServoingSkill`
- `ObjectTrackingSkill`
- `ObjectLocalizationSkill`
- `BoxDecompositionSkill`

- `CoViSGraspingSkill`
- `ObjectPushingSkill`
- `PlaceSkill`
- `ZeroForceSkill`
- `PouringSkill`
- `HandOverSkill`
- `GraspDoorHandle`
- `OpenDoorSkill`
- `CloseDoorSkill`
- `HumanMotionCaptureSkill`
- `GraspRecognitionSkill`
- `LearnDMPSkill`

In addition, a new communication architecture based on the Internet Communications Engine (ICE) was developed (KIT, UEDIN) and integrated into the ARMAR/ICE software framework, producing a significant improvement in the quality and speed of the plan generation process.

3. Integration of the PKS planner on the ARMAR robot platform

In work previously presented in deliverable D1.2.2, and developed as part of WP4 and WP5, we outlined the state of high-level planning on ARMAR, including the initial integration of the PKS planner into the ARMAR robot architecture. In the current reporting period we have extended this integration work in three ways: (i) the introduction of a new plan execution monitor component, (ii) the release of a new version of the PKS planning software, and (iii) a re-engineering of the planning domain specifications for the ARMAR kitchen domain.

The most significant development of the current reporting period has been the inclusion of a high-level plan execution monitor that operates in conjunction with the PKS planner. This component enables predicted states produced by the planner to be compared against observed states retrieved by the robot's vision system. At each plan step, the monitor assesses whether to continue executing an existing plan, resense particular objects in the world, or replan entirely. An initial version of the PKS plan execution monitor was previously integrated with SDU robot/vision system, and outlined in deliverable D4.3.5. However, a second version of this software is now available, and has been integrated into the ARMAR framework. More details about the plan execution monitor and its operation can be found in deliverable D5.1.3.

A new version of the PKS planning software has also been made available, featuring improvements to the quality and speed of the plan generation process. The library interface to PKS has also been reworked to integrate more smoothly with the plan execution monitor, and facilitate the distribution of future versions of PKS. In particular, the new plan execution monitor is now able to operate seamlessly with the latest version of PKS, and can take advantage of the new features available at the planning level. Again, more details about PKS and its extensions can be found in deliverable D5.1.3.

Finally, the current high-level planning domain description for the KIT kitchen environment (previously reported under WP4 as part of deliverable D4.3.5) has also been refined, to take into consideration the new capabilities of ARMAR, and changes in the physical kitchen environment. The most significant change has been the inclusion of three new actions that the planner can use for plan generation: the ability to grasp two objects at once (grasp-twoObjs), to put down two objects at once (put-down-twoObjs), and to pour liquid from a container into another object (pour).

The first two actions extend our previous work on generating plans with arm-specific operations (e.g., “grasp the cup with the left hand’’). The ability to grasp two objects simultaneously should also be seen as an intermediate step towards actions where a robot requires both hands in order to lift particularly large or awkwardly-shaped objects. The last action (pour) illustrates our first attempt at planning actions that require significant coordination between both hands. At the high level, the pour action also discretizes the continuous control problem of assessing fluid levels in an object, and simply considers objects to be “empty’’ or “full”. A more sophisticated version of this action is planned for the future, beyond the end of PACO-PLUS, as our planning and sensing capabilities improve.

The following table presents the current version of our high-level planning actions, described in PKS syntax. These actions replace the previous action specifications found in deliverables D4.3.5 and D5.1.3.

Actions	Preconditions	Effects

close(?l,?h)	((?l=cupboard & ?h=righthand) |
	(?l=dishwasher & ?h=righthand) |
	(?l=fridge & ?h=lefthand))
	robotLocation=?l
	(objOpen(?l) | objPartialOpen(?l)
	gripperEmpty(?h)
		del(Kf,objOpen(?l))
		del(Kf,objPartialOpen(?l))

grasp(?x,?l,?h)	object(?x)
	(?l=sideboard | ?l=stove)
	hand(?h)
	!flat(?x)
	!toppled(?x)
	robotLocation=?l
	objLocation(?x,?l)
	gripperEmpty(?h)
		add(Kf,inGripper(?x,?h))
		del(Kf,gripperEmpty(?h))
		del(Kf,objLocation(?x,?l))

grasp-fromEdge(?x,?l,?h)	object(?x)
	(?l=sideboard | ?l=stove)
	hand(?h)
	flat(?x)
	atEdge(?x)
	robotLocation=?l
	objLocation(?x,?l)
	gripperEmpty(?h)
		add(Kf,inGripper(?x,?h))
		del(Kf,gripperEmpty(?h))
		del(Kf,objLocation(?x,?l))
		del(Kf,atEdge(?x))

grasp-twoObjs(?x,1?x2,?l)	object(?x1)
	object(?x2)
	x1 != x2
	(?l=sideboard | ?l=stove)
		add(Kf,inGripper(?x1,lefthand)
		add(Kf,inGripper(?x2,righthand)
		del(Kf,gripperEmpty(lefthand))
		del(Kf,gripperEmpty(righthand))

	!flat(?x1)
	!flat(?x2)
	!toppled(?x1)
	!toppled(?x2)
	robotLocation=?l
	objLocation(?x1,?l)
	objLocation(?x2,?l)
	gripperEmpty(lefthand)
	gripperEmpty(righthand)
		del(Kf,objLocation(?x1,?l))
		del(Kf,objLocation(?x2,?l))

move(?l1,?l2)	location(?l1)
	location(?l2)
	?l1 != ?l2
	robotLocation=?l1
		add(Kf,robotLocation=?l2)

nudge-toEdge(?x,?l,?h)	object(?x)
	(?l=sideboard | ?l=stove)
	hand(?h)
	flat(?x)
	!atEdge(?x)
	robotLocation=?l
	objLocation(?x,?l)
	gripperEmpty(?h)
		add(Kf,atEdge(?x))

open(?l,?h)	(?l=cupboard | ?l=diskwasher) ?h=righthand robotLocation=?l !objOpen(?l) gripperEmpty(?h)	add(Kf,objOpen(?l))
open-partial(?l,?h)	?l=fridge ?h=lefthand robotLocation=?l !objOpen(?l) !objPartialOpen(?l) gripperEmpty(?h)	add(Kf,objPartialOpen(?l))

open-complete(?l,?h)	?l=fridge
	?h=righthand
	robotLocation=?l
	!objOpen(?l)
	objPartialOpen(?l)
	gripperEmpty(?h)
		add(Kf,objOpen(?l))
		del(Kf,objPartialOpen(?l))

pass-object(?x,?h1,?h2)	object(?x)
	hand(?h1)
	hand(?h2)
	?h1 != ?h2
	inGripper(?x,?h1)
		add(Kf,gripperEmpty(?h1))
		add(Kf,inGripper(?x,?h2))
		del(Kf,gripperEmpty(?h2))
		del(Kf,inGripper(?x,?h1))

	gripperEmpty(?h2)	

place-upright(?x,?l,?h)	object(?x)
	location(?l)
	hand(?h)
	toppled(?x)
	robotLocation=?l
	objLocation(?x,?l)
	gripperEmpty(?h)
		del(Kf,toppled(?x))

pour(?x1,?x2,?l,?h)	?x1=juice
	?x2=cup
	hand(?h)
	?l=sideboard
	robotLocation=?l
	inGripper(?x1,?h)
	objLocation(?x2,?l)
	empty(?x2)
		del(Kf,empty(?x2))
		add(Kf,full(?x2))

put-down(?x,?l,?h)	object(?x)
	(?l=sideboard | ?l=stove)
	hand(?h)
	robotLocation=?l
	inGripper(?x,?h)
		add(Kf,gripperEmpty(?h))
		add(Kf,objLocation(?x,?l))
		del(Kf,inGripper(?x,?h))

put-down-twoObjs(?x1,?x2,?l)	object(?x1)
	object(?x2)
	(?l=sideboard | ?l=stove)
	robotLocation=?l
	inGripper(?x1,lefthand)
	inGripper(?x2,righthand)
		add(Kf,gripperEmpty(lefthand))
		add(Kf,gripperEmpty(righthand))
		add(Kf,objLocation(?x1,?l))
		add(Kf,objLocation(?x2,?l))
		del(Kf,inGripper(?x1,lefthand))
		del(Kf,inGripper(?x2,righthand))

put-in(?x,?l,?h)	object(?x)
	((?l=cupboard & hand(?h)) |
	(?l=dishwasher & ?h=righthand) |
	(?l=fridge & ?h=lefthand))
	robotLocation=?l
	objOpen(?l)
	inGripper(?x,?h)
		add(Kf,gripperEmpty(?h))
		add(Kf,objLocation(?x,?l))
		del(Kf,inGripper(?x,?h))

remove-from(?x,?l,?h)	object(?x)
	((?l=cupboard & hand(?h)) |
	(?l=fridge & ?h=lefthand))
	robotLocation=?l
	objOpen(?l)
	objLocation(?x,?l)
	!toppled(?x)
	gripperEmpty(?h)
		add(Kf,inGripper(?x,?h))
		del(Kf,gripperEmpty(?h))
		del(Kf,objLocation(?x,?l))

At the time of reporting, software-level integration work is continuing, to incorporate all new PKS modules into the ARMAR framework. While this work has not in general been problematic, it has required a restructuring of some of the interface software on the planning side, to take advantage of the new ICE (Internet Communications Engine)-based communication architecture on the ARMAR platform. Future extensions to the planning capabilities of ARMAR are also scheduled to be implemented, beyond the end of PACO-PLUS.

Additional technical details about the integration of planning services on the ARMAR platform can be found in the document "*Integrating Low-Level Robot/Vision with High-Level Planning and Sensing in PACO-PLUS*", attached as part of deliverable D.5.1.3.