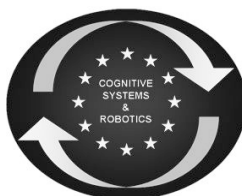




SAPHARI

SAFE AND AUTONOMOUS PHYSICAL HUMAN-AWARE ROBOT INTERACTION



Project funded by the European Community's 7th Framework Programme (FP7-ICT-2011-7)
Grant Agreement ICT-287513

Deliverable D7.3.1

An implemented human-aware manipulation planner

Report due date: 31 July 2014	Actual submission date: 30 September 2015
Start date of project: 1 November 2011	Duration: 48 months
Lead beneficiary for this deliverable: CNRS-LAAS	Revision: Final

www.saphari.eu

Executive Summary

This deliverable of workpackage WP7 reports on the availability of a software prototype of a human-aware manipulation planner, or more precisely of a comprehensive system allowing to plan at geometric level manipulation actions in a human environment. Such actions involve often several inter-related motions planning steps. The main constraint for a teammate robot is not only the feasibility of the task but also the human safety as well as acceptability and ease of legibility of the behavior of the robot by the human.

Table of contents

Executive Summary	1
Introduction	3
Planning human-aware manipulation	3
Software Integration framework	6
Software use and results	12
References	13

Introduction

This deliverable of workpackage WP7 reports on the availability of a software prototype of a human-aware manipulation planner, or more precisely of a comprehensive system allowing to plan at geometric level manipulation actions in a human environment. Such actions involve often several inter-related motions planning steps. The main constraint for a teammate robot is not only the feasibility of the task but also the human safety as well as acceptability and ease of legibility of the behavior of the robot by the human.

The next sections describe briefly the human-aware motion planner delivered and then the action-planning framework for synthesizing collaborative human-robot manipulation actions. We then report on its integration and use in SAPHARI. Detailed description of the algorithms is available in the published papers listed at the end of the document.

Planning human-aware manipulation

We use a T-RRT algorithm, which allows planning low-cost motions. The cost is engineered to create motions that seems more natural and comfortable to the human coworkers. It mainly concerns positions of the robot parts in the workspace, with respect to the human position.

The cost is actually a complex composition of several costs. The most important effects are maximizing the distance between the robot and its coworkers, and keeping the robot arm in a height range that corresponds to the human workspace (approximately between the human shoulders and hips). The table bellow shows the same planning query results with different cost to illustrate how they influence the motion. (The robot arm goes from above its platform to the bottle, in order to pick it)

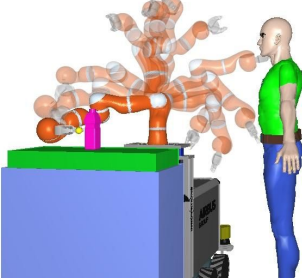
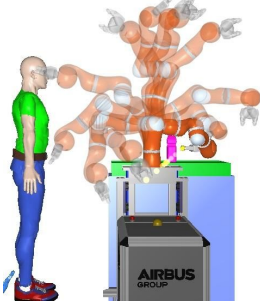
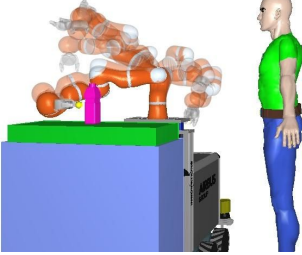
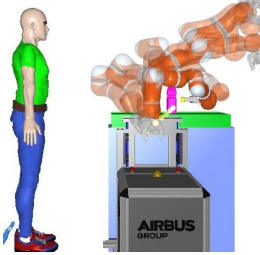
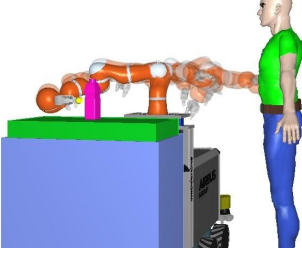
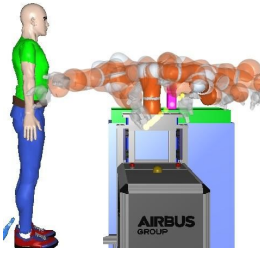
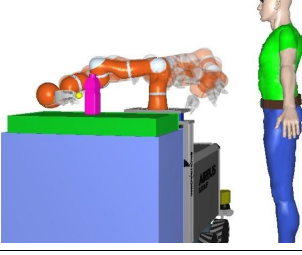
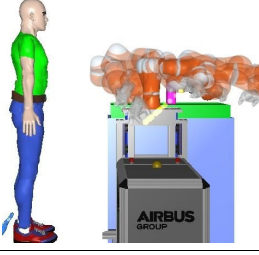
	Viewpoint 1	Viewpoint 2
No cost		
Distance cost		
Height cost		
Global cost		

Figure 1: Example of a planned path respecting a set of human-aware constraints (the robot is the one deployed in the AIRBUS Ariadne use case)

We use a bi-directional T-RRT-Extend variant. The *Extend* variant allows sampling more precisely the configuration-space and its cost, and so avoiding going through cost picks. We post-process the path generated by the motion planner in order to have a smoother motion. We use a random shortcut algorithm, which minimizes both robot parts displacement (in workspace, not in configuration-space) and the cost used for planning. We consider both the maximal cost along the path and the average cost to get good results.

During the manipulation planning workflow, this cost is also used to find good target positions among all the possible ones. For *pick* actions, it samples the grasp positions, for *place* actions, it samples the objects positions on the support before planning the motion. This way, the start and goal configurations given to the planner have a lower cost, which ease the planning and give better results in term of path cost. The above pictures show typical cases of a pick action with random and cost-based position selection. When the picking position is randomly selected, the T-RRT doesn't find a good plan, and the robot arm goes really close to the human torso. When the position is selected based on its cost, the motion is clearly better, the arms goes on the opposite side and stay far from the human.

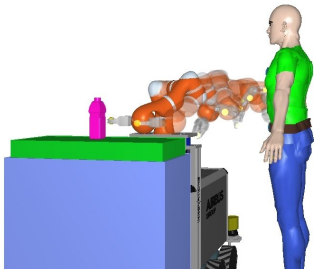
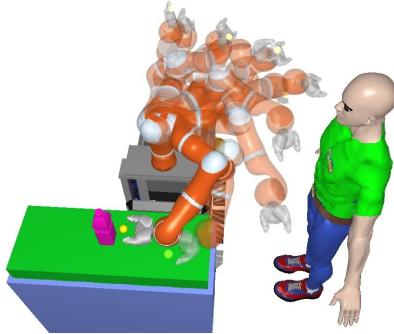
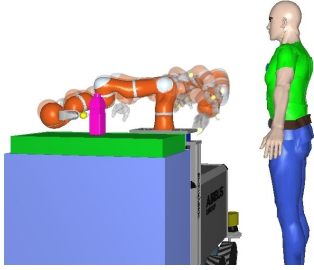
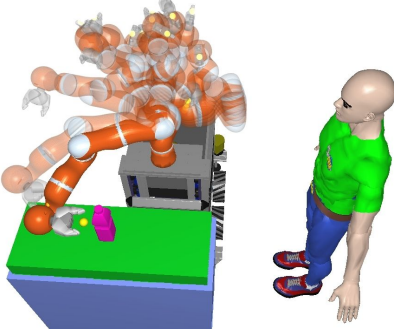
	Viewpoint 1	Viewpoint 2
Random position selection		
Cost-based position selection		

Figure 2: A typical example of a combination of motion and object grasp planning to ensure a better (in terms human-aware constraints) “Pick object” action

Software Integration framework

The human-aware manipulation planner has been integrated in a software framework called “Geometric Reasoning and Planning” framework (GRP) that has been specially designed to plan, at geometric level, collaborative human-robot actions. This is particularly useful since action planning generally involves several inter-dependent planning and selection steps: path planning, grasp and object placement planning, robot and human posture planning. GRP allows not only to plan human-aware actions but also to provide upon request several instances of the same action. Finally it provides a principled interface with higher level symbolic planning since it is able to refine actions, to provides several geometric instances of a same

action as well as to compute the effective side effects of actions in terms of affordances (see for instance Gharbi-et-al IEEE IROS 2015). In the sequel, we briefly describe how it has been implemented.

The goal of GRP framework is to create a link and fill the gap between the task planning and the motion planning. Usually, task planning manipulates symbols and concepts, and tries to find symbolic plans able to achieve a given goal while motion planning is used to compute the robot trajectories, which are executable in the real world, trajectories that enable it to achieve tasks. Motion planning is based on a geometric model of the world, and needs a full description of the initial and final position of the robot model. This full description is usually expressed with the numerical values related to the position of every part of the robot in the model.

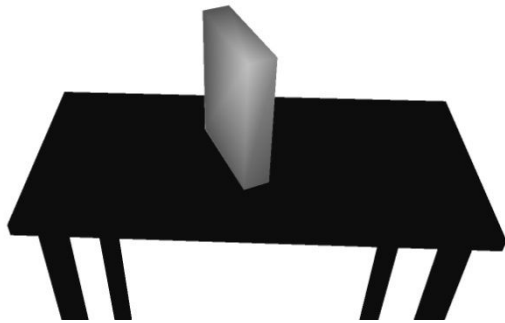
To synthesize, the task planning deals with symbols while motion planning requires specific numerical values to compute the trajectories. The gap between these two planners is the problem we are trying to solve in the context of manipulation and navigation planning, using fetch and carry examples in the presence and in interaction with humans. Let's refer to this problem as the Geometric Reasoning and Planning (GRP) problem.

The main goal of the GRP is to compute actions: Based on symbols, the GRP should compute trajectories, positions, rotations and grasps (when needed) that will achieve the goals specified at an abstract level. In other words, it should be possible to plan for actions while specifying only the desired information: the desired property to achieve at a level of abstraction sufficiently high to be usable by the task planner. For example, giving an object to this person or putting an additional object on the table. This is even more important when other (human oriented) constraints have to be taken into account. The GRP can have another usage, which is to compute Facts, based on the geometric model of the world; it is able to compute symbols describing the actual world state. We call these symbols facts. For example, it should be able to compute facts such as an object is in another one, an object is on another one, or more human related ones such as an object is reachable by an agent. These links between agents and objects are called affordances.

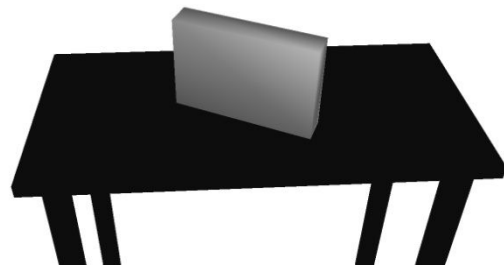
When requested to compute an action, the GRP framework uses the following to find a solution: First it computes a final world state. A world state is the position and configuration of every robot and object in the world. In order to compute this final state, the framework uses different tools and information:

- First, the framework finds a position for the manipulated objects (when relevant), using if needed the fact and affordances it is able to compute. For example, one action that the framework is able to compute is to place an (already in hand) object on a table such as it is reachable by a human. In

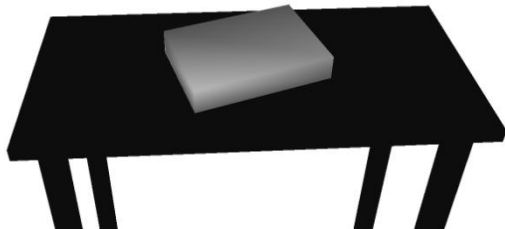
this case, the framework finds a position and a rotation for the object such as it is stable on the table surface (see Fig. 3) and the human is able to reach it from his actual position by only moving his arm. For a Pick this step is not needed, but a Grasp is chosen among the collision free ones and used for the next step.



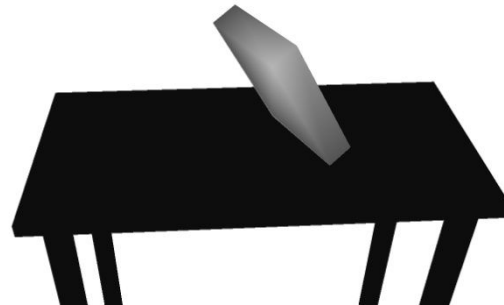
A



B



C



D

Figure 3: A, B and C are different stable configurations for the grey book (this is just a sample from the available stable configurations). D is not a stable configuration.

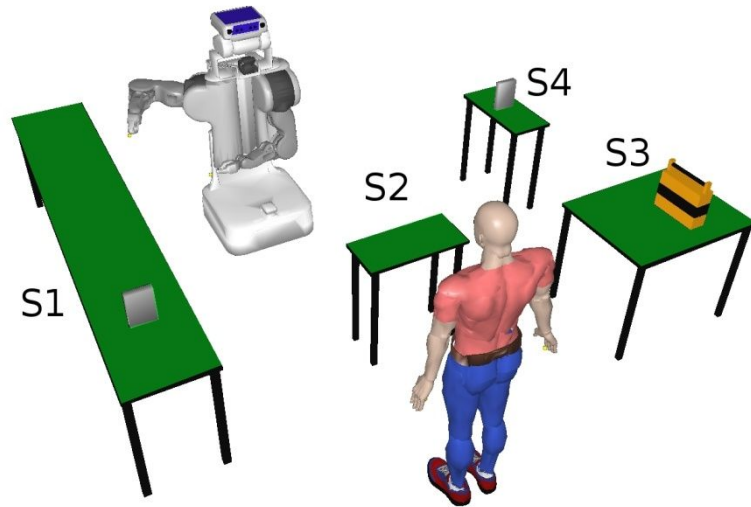


Figure 4: The supports of the tables are represented in green, each table has one support, which is a rectangle covering its top face. The objects on the tables are not support objects; hence, they don't have any supports.

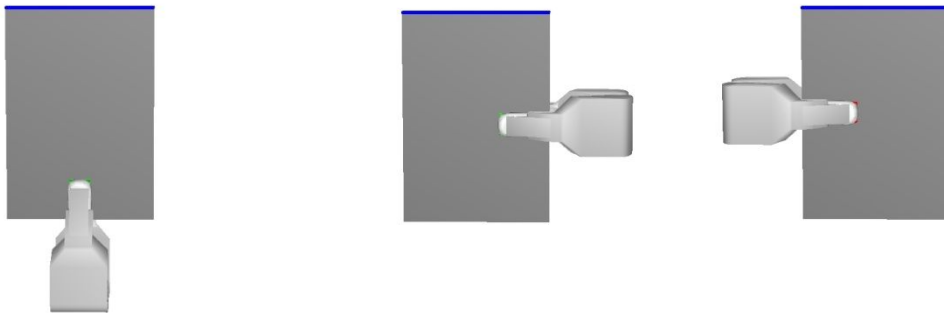


Figure 5: Different grasps for the grey book (this is just a sample from the available grasps).

- Then, the GRP framework computes the robot position using inverse kinematics (IK): in the case of the place reachable, once the object position is found, the robot end effector position is inferred from the known grasp of the object, then the arm position is computed with the IK.
- Finally, the trajectory is computed between the initial position of the robot and the final one that the framework just computed (Figure 6).

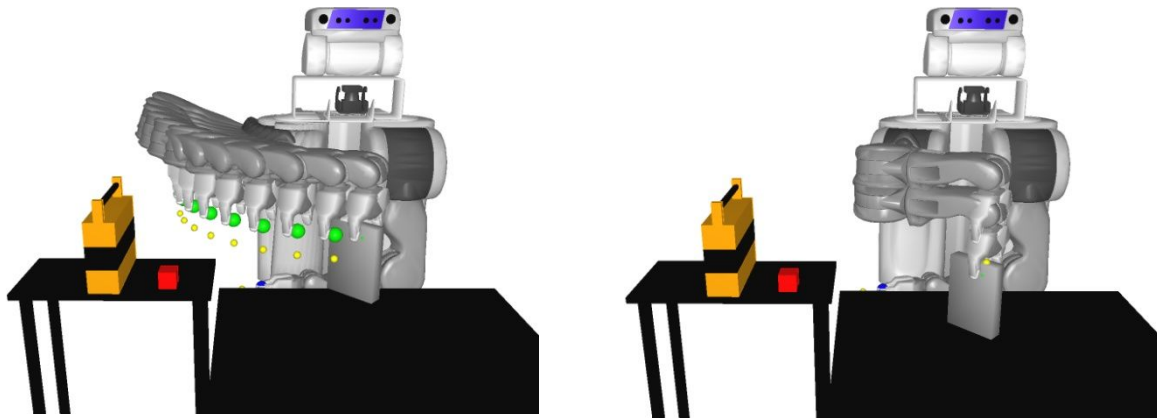


Figure 6: Approach and place an object

Various actions are now available in the GRP framework: **Pick**, **Place**, **Place Reachable**, **Drop**, **Stack**, and **Navigate To**.

In order to evaluate the framework capacities, some tests were held with the actions **Pick**, **Place** and **Place Reachable**. These results have been evaluated in a scenario where a PR2 robot needs to Pick (or **Place**, or **Place Reachable** to a human in the other side of the table) a green bottle, with one of its two 7 Degree of Freedoms arms, on a table in front of him. Some initial world states are depicted in Figure 7 (the robot arm and the bottles initial configuration have been randomized, the figure shows only some examples of this initial scenarios for a **Pick**). The motion planner used is a bidirectional balanced RRT, and the test was run on an Intel Xeon® CPU W3520 @ 2.67GHz x 4, on one core only.



Figure 7: Various initial world states where the Pick has been evaluated

Table 1: Time means the computation time, Sol Tests means the number of solutions explored (by how much solLeft decreased), Inverse kinematic means the number of inverse kinematic called, and Motion plan means the number of calls to the motion planner. These averages, variance and standard deviation (stand dev) are computed in over 150 successful action computations.

for one action	without motion plan			with motion plan		
Pick	average	variance	stand dev	average	variance	stand dev
Time	0.026	0.0001	0.0108	2.8553	17.1426	4.1403
Sol tests	2.525	3.6193	1.9024	8.2130	124.558	11.1606
Inverse kinematic	4.61	4.5379	2.1302	11.4556	128.899	11.3534
Motion plan	-	-	-	2.0532	2.1687	1.4726
Place						
Time	0.0201	0.0007	0.0270	2.7153	22.8922	4.7845
Sol tests	4.4522	19.7352	4.4424	18.5033	1166.78	34.1582
Inverse kinematic	4.9296	7.3216	2.7058	11.7219	217.101	14.7344
Motion plan	-	-	-	2.0463	2.4548	1.5667
PlaceReachable						
Time	0.0477	0.0016	0.0403	3.0798	47.1862	6.8692
Sol tests	5.5577	78.4879	8.8593	12.2692	236.735	15.3862
Inverse kinematic	5.1658	10.5303	3.2450	9.4359	57.8741	7.6075
Motion plan	-	-	-	1.8846	1.8969	1.3773

Table 1 is divided into two parts; the left one is the computation without motion plan and the right one with motion plan. This separation is meant to show the framework performance decoupled from the motion planner performances, as the one used can be exchanged with any off-the-shelf one. The first line shows each action mean computation time. The first interesting aspect is that the motion planning takes most of the computation time. Another interesting aspect is that **Place Reachable** needs more computation time than the **Place** as it incorporates an additional constraint and takes more time to account for it. One can also notice that the number of solutions explored in the part without motion plan is significantly smaller than the one with the motion plan: the algorithm fails often to find trajectories, which is reflected in the two last parameters, the number of calls to the inverse kinematic solver is bigger in the right side and the mean calls number to the motion planner is ~ 2 with a variance ~ 2 . During the motion planning, most of the examples were very fast to compute (as shown by the low averages of the computation times) but in very few examples, the motion planning took a long time, making the variance and the standard deviation very high. One particular number to look for in the table is the variance of the number of solutions explored in the case of a **Place**: this high number can be explained by the number of variable the **Place** action needs to instantiate in order to find a solution.

The framework is also able to handle a series of action, organized as a geometric plan: it is able to compute a new action based on the final world state of another action, making the links between the action clear. Moreover, for each action, as there might be multiple solution (for example, an object can be placed in

multiple position on a table), the framework is able to compute different alternatives for the same action. Figure 8 shows a geometric plan with some alternative in some actions.

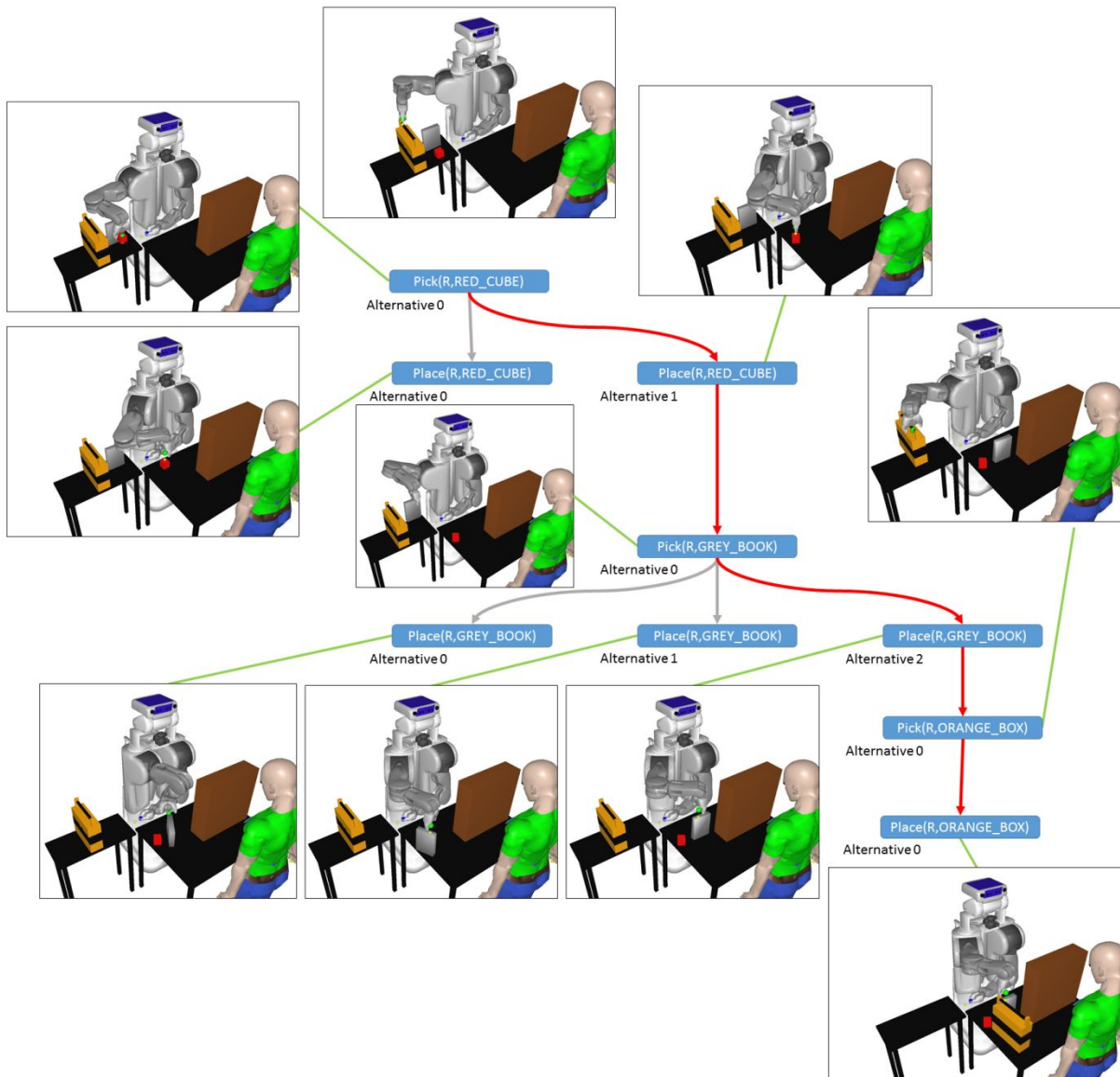


Figure 8: The different steps of a geometric plan, including the unused alternatives. In the plan, the robot performs three successive Pick and Place on three objects.

Software use and results

As mentioned above, the software is delivered and available for use in the framework of SAPHARI. It has been deployed and tested quite intensively in different conditions and with different robots: PR2s in the AIRBUS use case mock-up deployed at LAAS, AIRBUS use case mobile manipulator deployed in T8.3.

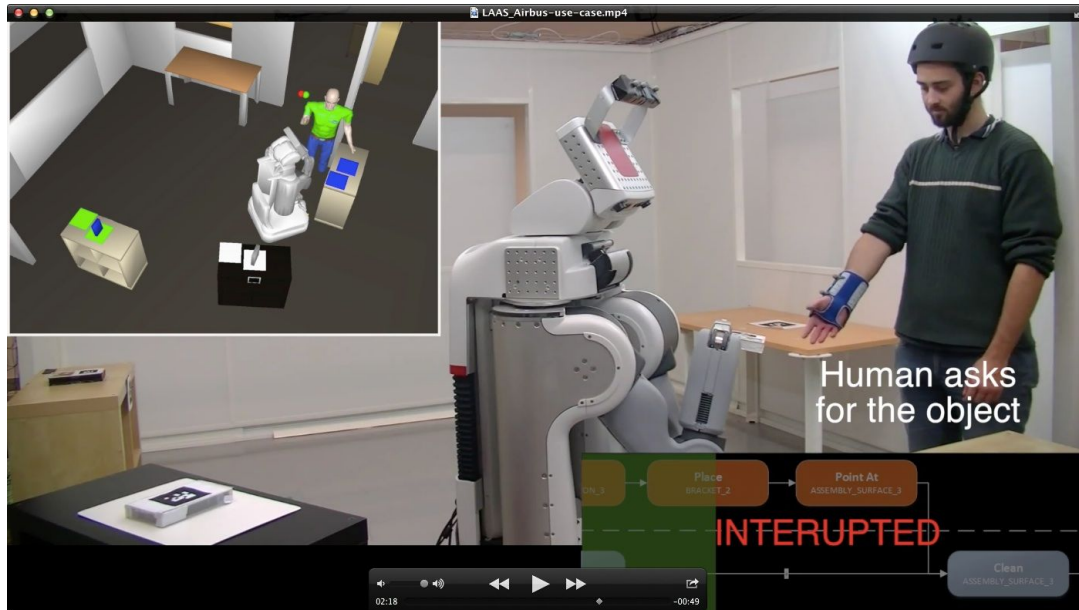


Figure 9: Mock-up of the AIRBUS Ariadne use case including all decisional components developed in WP7. The human-aware motion and task planning system is used in combination with a high-level symbolic planner to synthesize human-robot shared plans.

References

Journal papers

A Human-Aware Manipulation Planner. E.A.K. Sisbot, Rachid Alami, IEEE Transactions on Robotics, ISSN: 1552-3098, Digital Object Identifier: 10.1109/TRO.2012.2196303, 2012.

Towards Human-Level Semantics Understanding of Human-Centered Object Manipulation Tasks for HRI: Reasoning About Effect, Ability, Effort and Perspective Taking, Amit Kumar Pandey and Rachid Alami, “, International Journal of Social Robotics (IJSR) 6(4): 593-620 (2014)

Ingredients and a Framework of Dexterous Manipulation Skills for Robots in Human Centered Environment and HRI, Amit Kumar Pandey and Rachid Alami, Journal of Robotics Society of Japan, Volume 32, No. 4 (RSJ-2014).

Conference papers

Lavindra de Silva Amit Kumar Pandey Rachid Alami, "An Interface for Interleaved Symbolic-Geometric Planning and Backtracking", IEEE/RSJ International Conference on Intelligent Robots and Systems, November 3-8, 2013, Tokyo, Japan

Lavindra de Silva Amit Kumar Pandey Mamoun Gharbi Rachid Alami, "Towards Combining HTN Planning and Geometric Task Planning", Workshop on "Combined Robot Motion Planning and AI Planning for Practical Applications", Robotics: Science and Systems 2013, Berlin, June 2013.

Amit Kumar Pandey and Rachid Alami, "Affordance Graph: A Framework to Encode Perspective Taking and Effort based Affordances for day-to-day Human-Robot Interaction", IEEE/RSJ International Conference on Intelligent Robots and Systems, November 3-8, 2013, Tokyo, Japan

On human-aware task and motion planning abilities for a teammate robot. Rachid Alami, Mamoun Gharbi, Benjamin Vadant, Raphael Lallement and Adolfo Suarez. In Proc. of WS RSS 2014.

Raphael Lallement, Lavindra de Silva, and Rachid Alami. HATP: An HTN Planner for Robotics. In 2nd ICAPS Workshop on Planning and Robotics, PlanRob 2014, 2014.

Lavindra de Silva, Amit Kumar Pandey and Rachid Alami, Towards a Principled Approach to Symbolic-Geometric Planning, IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2013).

Lavindra De Silva, Mamoun Gharbi, Amit Kumar Pandey and Rachid Alami, A New Approach to Combined Symbolic-Geometric Backtracking in the Context of Human-Robot Interaction, IEEE International Conference on Robotics and Automation (ICRA 2014).

Amit Kumar Pandey and Rachid Alami, Affordance Graph: A Framework to Encode Effort-based Affordances for day-to-day HRI, IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2013).

Mamoun Gharbi, Raphael Lallement, Rachid Alami, Combining Symbolic and Geometric Planning to synthesize human-aware plans: toward more efficient combined search. IEEE IROS 2015

Background publications of the team used here but not published under SAPHARI

Emrah Akin Sisbot, Luis Felipe Marin-Urias, Xavier Broquère, Daniel Sidobre, and Rachid Alami: Synthesizing Robot Motions Adapted to Human Presence - A Planning and Control Framework for Safe and Socially Acceptable Robot Motions. I. J. Social Robotics 2(3): 329-343 (2010)