

Concept of a Hybrid Architecture for Care-O-bot

Matthias Hans, Winfried Baum

Fraunhofer Institute for Manufacturing Engineering and Automation (IPA)

Nobelstrasse 12, 70569 Stuttgart, Germany

e-mail: {hans, baum}@ipa.fhg.de

Abstract

Care-O-bot is the prototype of a multi-functional robot assistant for housekeeping and home care, to be used by elderly people in order to live independently in their homes for a longer time. Therefore, an easy, intuitive, and dependable operation of the home care system is essential.

Care-O-bot has to cope with many different situations and has to fulfill complex tasks even in dynamic environments. Furthermore, the robotic assistant must be able to execute not only single tasks at a time but several tasks concurrently. In this paper we present the concept for hybrid control architecture appropriate for performing different tasks over long periods of time.

Furthermore we show our experience with JAM on the robot assistant rob@work, which was presented successfully at the Hannover Trade Fair 2001 and conclude with the need of a symbolic planner for robotic assistants.

Key words: hybrid control architecture, task planning, robot assistant for housekeeping and home care



Figure 1. Intelligent Home Care System Care-O-bot

1 Introduction

There are different motivating factors for the employment of robot assistants [6] for housekeeping and home care: on the one side, comfort factors and a changing societal framework; on the other side, an increasing number of households include inhabitants that require physical support in daily life due to sickness or age.

By the year 2030 the number of people who are 60 years old will have doubled in Germany [15]. Technical aids are required to allow people to live independently and supported in their private homes as long as they wish. As a contribution a demonstrator platform for a mobile home care system – called Care-O-bot (Figure 1) – was designed and realized by Fraunhofer IPA [17].

Care-O-bot II, the next generation prototype, is currently under construction. The new platform will have a manipulator to perform tasks such as fetch-and-carry duties, setting the table or basic cleaning. The robot assistant will have to be able to move through the various rooms of the home without colliding with furniture or people. A basic understanding of the structure and dynamics of the surrounding environment and its operational context is fundamental for its goal-directed behavior. The robot must be able to analyze and interpret events in the surrounding world in order to react appropriately.

The foremost needs of an appropriate control system are to be able to achieve high-level complex goals, to interact with a complex, often dynamic environment, to be able to ensure the system's own dynamics, to handle stochastic perturbations and uncertainty, and to be reactive to unexpected changes. Several tasks must be executed concurrently, and the ability to interrupt these tasks when desired must also exist.

2 Concept of System Architecture

A good overview on robot architectures is given in [2]. For optimal performance of the given tasks we have

chosen an architecture classified as a hybrid. It combines both reactive and deliberative control in a heterogeneous architecture.

The basic modules of the new system architecture of Care-O-bot II are shown in Figure 2. Regular arrows represent data flow, dashed arrows represent control flow.

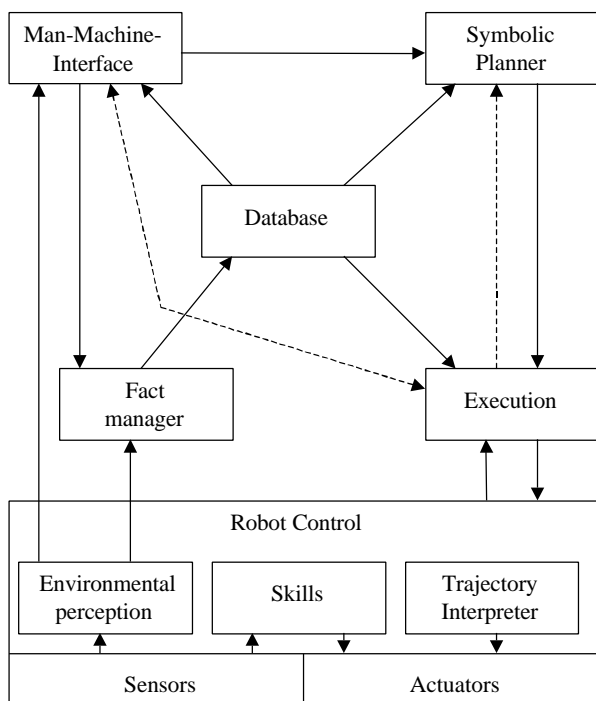


Figure 2. Basic modules of the system architecture of Care-O-bot II

Communication between human and robot assistant via the *man-machine-interface* drive task execution. Orders are sent to the *symbolic planner*, which generates a list of possible actions that could be used to reach the goal. The *execution module* picks a suitable action from the list and triggers the *robot control* to execute it. All modules need a world-model containing information about the environment, which is available from the *database*. With its sensors the robot assistant analyses its environment and updates the database. Further, it displays the progress of task execution via the man-machine-Interface.

The modules of the system architecture are explained in detail in the following:

2.1 Man-Machine-Interface

The Man-Machine-Interface is the user interface between human and robot assistant. Care-O-bot receives orders via the input channels for speech and touch-screen and converts them into commands for the symbolic plan-

ner. The robot assistant displays its current state. The human monitors task execution and can interact at any moment.

The primary user interface includes the graphical interface, used for visualizing the robot and user's environment as well as a speech in/output system. The user control panel displays all current variables of the robot, as e.g. the motion state, the position, or planned actions (e.g. planned path) of the robot, state of the speech input system, and possible error messages. An online help system is also provided.

Important components of the user interface include a fusion module, which merges all information given through the communication channels, and a separation module, which relates the output on the adequate channels. The module gets context-dependable data for output such as sound files, vrmf-files, etc. from the database.

2.2 Database

The database is a structured collection of data. It contains all the data about the environment of the robot assistant. This data includes maps and information about the objects in the environment:

- name
- properties (mobile, open/closed,...)
- relations (is part of, lays on,...)
- geometric identification (surface, size,...)
- specific information for sensors how to identify certain objects
- data how to grip certain objects
- methods, what to do with the object
- last known position
- visualization information for the user interface

For software implementation the relational database management system MySQL [14] has been chosen. This system stores data in separate tables, which are linked by defined relations making it possible to combine data from several tables on request. SQL (Structured Query Language), the most common standardized language, is used to access the database. MySQL is Open Source Software and could be modified to fit our needs. It is fast, compact and easy to use.

2.3 Symbolic Planner

The symbolic planner generates a list of actions to reach the goals specified by the user.

Development of symbolic planners has been a central topic in Artificial Intelligence (AI) for decades. A good overview is given in [8]. For Care-O-bot we chose a planner based on ADL (Action Description Language [15]).

AI-planners compete every second year on the International Conference of Artificial Intelligence Planning &

Scheduling (AIPS) [12]. The winner of the last competition in 2000 [1] in ADL-category was FF [9][10]. Some sample problems for Care-O-bot tested on FF were solved in less than a second.

The FF-planner uses the Planning Domain Description Language (PDDL [16]), which is the standardized language for all participants of the planner competition. This requires converting data from the man-machine-interface to PDDL. The option to exchange the planner is an advantage because of its standardized interface.

The disadvantage of the state of the art planners for the competition is that they do not consider the costs of energy or time. We expect that the next generation of symbolic planners will be improved in that perspective.

The symbolic planner needs a closed-world domain description in PDDL-format. This description must contain a world-model and a list of the abilities of the robot. The planner can send requests to the database, where this information is saved. A filter to compose the right domain from the database is necessary.

2.4 Execution Module

For executing the generated plan we use JAM Agents [11]. This software-package is a BDI-theoretic (Believe-Desire-Intention) agent architecture based on the Procedural Reasoning System (PRS) of SRI International [4]. After specifying beliefs (facts known to the agent), desires (goals that the agent is to achieve) and capabilities (plans and primitive actions), intentions are determined dynamically by the agent at runtime, based on known facts, current goals and available plans.

JAM supports both top-down, goal-based reasoning and bottom-up data-driven reasoning. It selects goals and plans based on maximum priority, considering the preconditions, runtime attributes and postcondition attributes.

Facts and capabilities are provided by the database. The goals are the list of actions calculated by the symbolic planner. JAM keeps track of the progress the agent has made toward accomplishing those goals. The progress is shown to the human via the man-machine-interface. The user can confirm or modify plan-execution.

JAM agents explicitly handle failures caused during execution. Depending on the failure, the agents trigger the symbolic planner to find a new solution to the new situation. If the planner can not find any solution the user gets invoked via the man-machine-interface. The user can give new commands or new facts to the robot assistant to support the planner in finding a solution to accomplish the given goals.

2.5 Robot Control

The robot control is based on the *Realtime Framework* (RTF), developed at Fraunhofer IPA [19]. It is a

package of software modules for building distributed real-time control systems in robotics and automation. The *Realtime Framework* covers the areas of client-server communication, control of program flow and modality through messaging and state machines, and low-level input/output. It is primarily aimed at increasing software modularity, portability, and re-usability, thereby reducing software development costs.

Some of the components represent skills, which are closed-loop controllers. Examples include the velocity controller, position controller or the collision avoidance. Furthermore, we have open-loop controllers e. g. an interpreter for pre-programmed trajectories for the manipulator.

Another important part of robot control is environment perception. In addition to supporting sensory data, it also includes the signal processing and interpretation as e. g. the map-builder or the object identification and localization.

The pre-processed environment perception data is sent to the man-machine-interface in order to give the user information about the robot's surrounding. Furthermore, the progress of task execution of the robot is monitored. The environment data is also sent to the fact manager to keep the world-model up to date.

2.6 Fact Manager

The fact manager is responsible for keeping the world-model of the database up to date. It receives the pre-processed data of the environment perception and generates facts for the planner domain and plans for the JAM agents. Furthermore, it is responsible for confirming the knowledge of the surrounding environment, e.g. if the planner falsely believes that there is a cup to fetch when there is not because it had been put away, the execution of the JAM plan fails. The fact manager continuously updates the world-model in order to accomplish given tasks.

The second task of the fact manager is to build an interface between the database and the man-machine-interface. The user can give information to the robot: it can give statements, which get converted into PDDL- and JAM-facts.

3 Experience

The first mobile platform Care-O-bot was built at Fraunhofer IPA in 1998. Its primary task was save navigation in home environments.

The following robots based on the same hardware platform have been installed in March 2000 for the "Museum für Kommunikation Berlin" where they autonomously move among the visitors, communicate to and

interact with them [5][18]. Even though every robot might move in reaction to its environment, their task is fixed.

Care-O-bot II (Figure 3) will be equipped with a manipulator arm and walking supporters. These additions enable new tasks such as fetch-and-carry or other handling tasks, which require the system architecture concept presented.

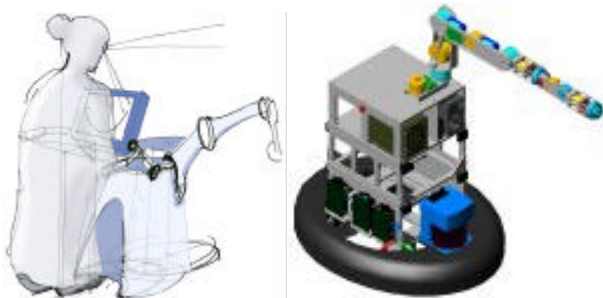


Figure 3. Care-O-bot II design (March 2001) and CAD model of mobile manipulator [3]

The newest development of Fraunhofer IPA, the robot assistant “rob@work” [7] is a related platform. The first prototype was successfully displayed at the Hannover Trade Fair 2001. It is based on the same hardware- and software-technology for the mobile platform as the Care-O-bots and the museum robots. In addition, a Mitsubishi PA-10 manipulator (see Figure 4) has been mounted. When developing the software architecture for this platform we used the JAM agents rather than a symbolic planner for executing tasks.

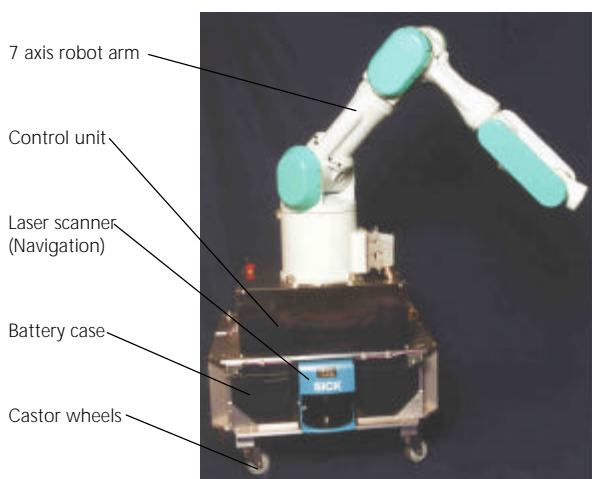


Figure 4. Manufacturing assistant rob@work [6] without its cover and interfaces

The presented scenario was part of the assembly of hydraulic pumps. Rob@work fetched shafts from where

they were stored and carried them to a manual workplace. The worker took them and assembled the shafts to the casing. The process of these shafts being handed over by a worker to the gripper of rob@work is also shown (Figure 5). The robot then carried the shafts back to the storage location.

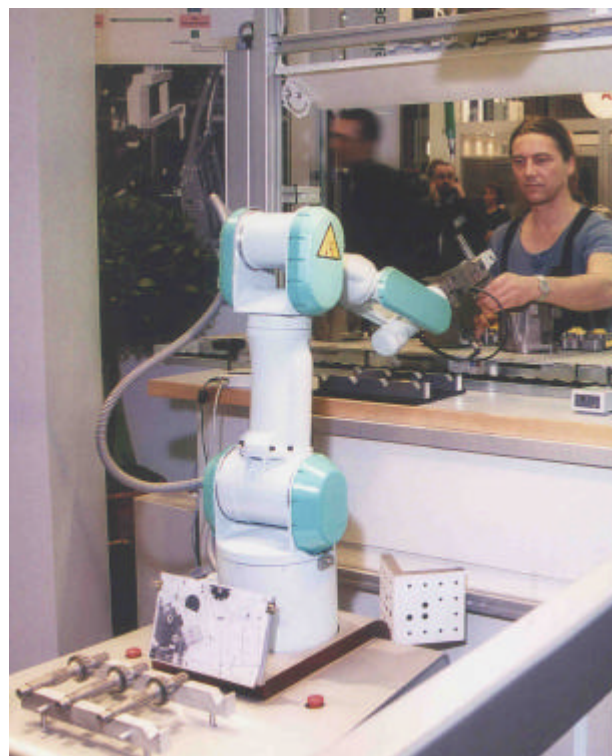


Figure 5. Hand over of a shaft from the worker to the manufacturing assistant rob@work. [Hannover Trade Fair 2001]

This scenario was realized with our Robot Control based on the Realtime Framework and the JAM agents for execution. The two components are connected via CORBA middleware. JAM agents did not only control the task execution and achievement of the goals, they also controlled the movement of the arm. We received positive experience in using JAM: after implementing the software package, it was very easy to change the scenario. For example, to change the arm-movements we only needed to define a new position and add a new possible action to the JAM facts list. The JAM system used the new facts to achieve given goals automatically.

4 Outlook

Although changing facts and adding possible actions is very easy, we will not continue to use JAM for the manipulator control. It was too slow to recognize failures

and to stop the system within a suitable amount of time. Our future robot control will have its own manipulator control, which will be triggered by the JAM agents. Failures will be handled directly from that local control and only error messages will be transmitted to JAM.

JAM is not able to plan in advance. It executes the first programmed action that fulfills the preconditions. Planning in advance is important for execution of concurrent tasks. Therefore, we require a symbolic planner. FF is able to plan in advance and generate plans to achieve interlocked goals.

5 Summary

We presented a concept for a hybrid control architecture consisting of a man-machine-interface, a symbolic planner, an execution module, the robot control and a database with a fact manager to keep the database up to date. As planner we use FF, a fast AI-planner, which provides ADL. For the execution module we use JAM agents, which is based on PRS. The database is built with MySQL.

Symbolic planners are necessary for robotic assistants to plan in advance and achieve interlocked goals. The execution module adds a reactive component to react to changes in the dynamic environment.

6 Acknowledgements

Part of this work has been supported by the MORPHA-project [13] funded by the German Ministry of Education and Research (bmb+f) under grant 01IL902G/9.

7 References

- [1] AIPS-00 Planning Competition: *The Fifth International Conference on Artificial Intelligence Planning and Scheduling Systems*, Breckenridge, CO, April 15-19, 2000, <http://www.cs.toronto.edu/aips2000/>.
- [2] Coste-Manière, Ève; Simmons, Reid: "Architecture, the Backbone of Robotic Systems". In *Proceedings of the 2000 IEEE International Conference on Robotics & Automation*, San Francisco, CA, pp. 67-72, April 2000.
- [3] Genrob Homepage: <http://www.genrob.de>, 2001.
- [4] Georgeff, M. P.; Lansky, A.: "Reactive Reasoning and Planning". In *Proceedings of AAAI-87*, pp. 677-682, 1987.
- [5] Graf, B.; Schraft, R. D.; Neugebauer, J.: "A Mobile Robot Platform for Assistance and Entertainment." In *Proceedings of ISR-2000*, Montreal, pp. 252-253.
- [6] Hägele M., Schraft, R.D.; Neugebauer, J.: "From Robots to Robot Assistants". In *Proceedings of ISR-2001*, pp. 404-409, 2001.
- [7] Helms, E.; Hägele, M.: "rob@work – the helping hand". Topic 4 in: *Research News No. 4-2001*, Fraunhofer Gesellschaft, München, 2001.
- [8] Hertzberg, J.: "Planning". In *Encyclopedia of Electrical and Electronics Engineering (J. Webster, ed.)*, Wiley, 1999.
- [9] Hoffman, J.: FF-Homepage: <http://www.informatik.uni-freiburg.de/~hoffmann/ff.html>.
- [10] Hoffmann, J.; Nebel, B.: "The FF Planning System: Fast Plan Generation Through Heuristic Search". In *Journal of Artificial Intelligence Research*, Vol. 14, pages 253-302, 2001.
- [11] Huber, M. J.: "JAM Agents in a Nutshell", Version 0.61 + 0.79i.; July 4th, 1999.
- [12] *International Conference on Artificial Intelligence Planning & Scheduling Systems*, <http://www.isi.edu/aips>, 2001
- [13] "MORPHA – intelligente antropomorphe Assistenzsysteme", <http://www.morpha.de>, 2001.
- [14] MySQL AB Homepage, <http://www.mysql.com>, 2001.
- [15] Pednault, E. P. D.: "ADL: Exploring the middle ground between STRIPS and the situation calculus". In *Proceedings of the International Conference on Principles of Knowledge Representation (KR-98)*, pp. 324-332, 1989.
- [16] PDDL-manual available at: <http://www.informatik.uni-freiburg.de/~koehler/aips/PDDL-MANUAL.ps.gz> or <ftp://ftp.cs.yale.edu/pub/mcdermott/software/pddl.tar.gz>
- [17] Schaeffer, C.; May, T.: "Care-O-bot: A System for Assisting Elderly or Disabled Persons in Home Environments". In *Proceedings of AAATE-99*, Düsseldorf, 1999, pp. 340-345.
- [18] Schraft, R. D.; Graf, B.; Traub, A.; John, D.: „A Mobile Robot Platform for Assistance and Entertainment". In *Industrial Robot Journal*, Vol. 28, pp. 29-34, 2001.
- [19] Traub, A.; Schraft, R. D.: "An Object-Oriented Real-time Framework for Distributed Control Systems", in *Proceedings of ICRA-99*, pp. 3115-3121, 1999.