IEEE/RSJ Intern. Conf. on Intelligent Robots and Systems (IROS 2004), Japan, Sep. 28 - Oct. 2, 2004

A Modular and Distributed Embedded Control Architecture for Humanoid Robots

D.N. Ly, K. Regenstein, T. Asfour and R. Dillmann
Research Center for Information Technologies (FZI)
Haid-und-Neu-Str. 14, D-76131 Karlsruhe, Germany
http://www.fzi.de
ly@fzi.de, regenstein@fzi.de, asfour@ira.uka.de, dillmann@ira.uka.de

***Abstract*— In this paper we present a modular and distributed control architecture in order to achieve natural interaction and mobile manipulation task goals for a humanoid robot. We propose a hierarchically organized architecture with three levels and introduce the mapping of the functional features in this architecture into hardware and software modules. We also describe different functional features which have been realized and integrated into the whole control architecture.**

I. Introduction

The term humanoid is commonly associated with the idea of robots whose physical appearance is similar to that of the human body. Beyond a physical resemblance, humanoid robots must resemble humans in their ways of acting in the world, of reasoning and communicating about the world.

Our current research interest is the development of a control architecture to achieve manipulation task goals for a humanoid robot. The control architecture must provide the possibility to integrate the motor, perception and cognition components necessary for natural multi-modal human-humanoid interaction and human-humanoid cooperation. In particular, we address the programming and coordinated execution of manipulation tasks in a household environment. Therefore, it is an important issue to coordinate the multiple subsystems of a humanoid robot in carrying out tasks in dynamic and unstructured environments.

The design of humanoid robots requires coordinated and integrated research efforts that span a wide range of disciplines such as learning theory, control theory, artificial intelligence, human-machine-interaction, mechatronics, perception (both computational and psychological) and even biomechanics and computational neuroscience. These fields have usually been explored independently, leading to significant results within each discipline. The integration of these disciplines for the building of adaptive humanoid robots requires enormous collaborative resources that can be achieved only through a long-term, multidisciplinary research projects, as the German humanoid research project (SFB 588) initiated by the German Research Foundation (Deutsche Forschungsgemeinschaft: DFG). In the framework of the project we are working on the building and integration of humanoid robot components.

In this paper we present our ongoing work on the realization of a humanoid robot platform for household environments. In particular, we address the programming and coordinated execution of manipulation tasks in a household environment. Therefore, it is an important issue to coordinate the multiple subsystems of a humanoid robot in carrying out tasks in dynamic unstructured environments.

Fig. 1. The humanoid robot ARMAR

The paper is organized as follows. In section II a short description of the humanoid robot ARMAR is given. Section III describes the proposed control architecture including its hardware and software modules. The implemented features are presented in section IV. Finally, section V summarizes the results and concludes the paper.

II. The Humanoid Robot ARMAR

The humanoid robot ARMAR (see figure 1) has 23 mechanical degrees-of-freedom (DOF). From the kinematics

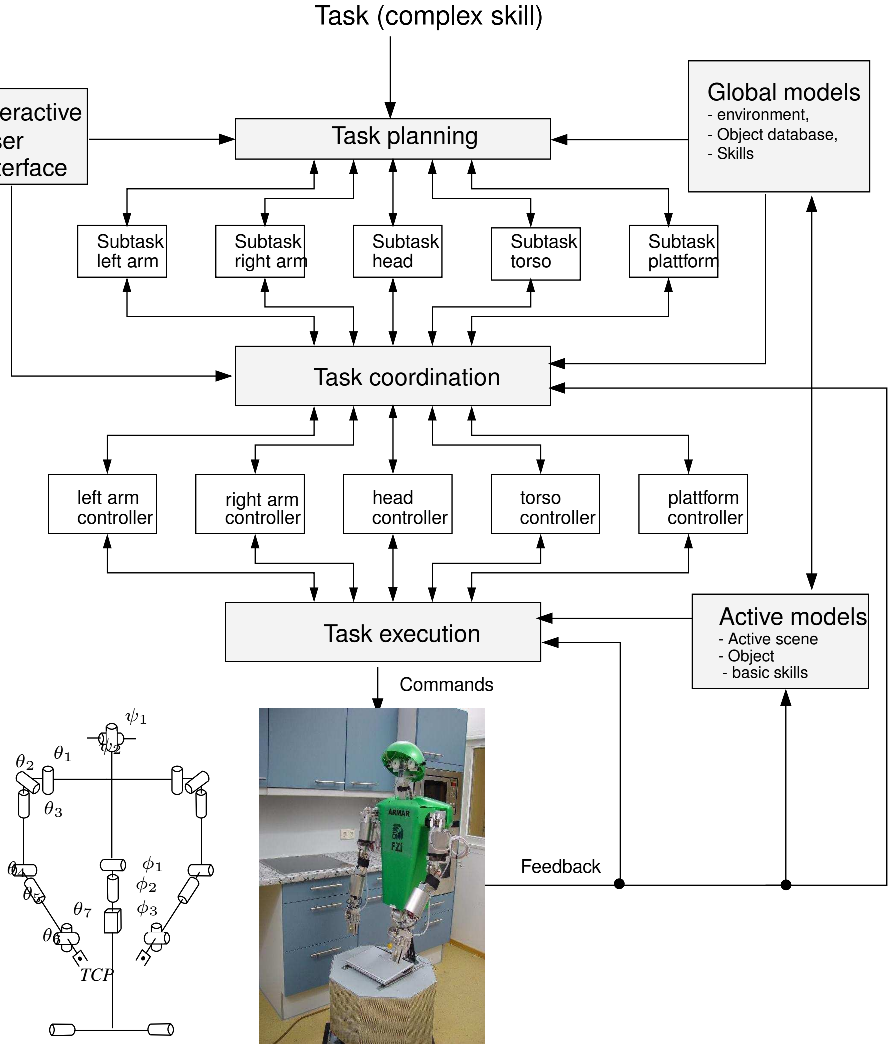

Fig. 2. The proposed control architecture with three levels: task planning, task coordination and task execution

control point of view, the robot consists of five subsystems: Head, left arm, right arm, torso and a mobile platform. The upper body of ARMAR has been designed to be modular and light-weight while retaining similar size and proportion as an average person. The head has 2 DOFs arranged as pan and tilt and is equipped with a stereo camera system and a stereo microphone system. Each of the arms has 7 DOFs and is equipped with 6 DOFs force torque sensors [12] on the wrist. The current mobile base of ARMAR consists of a differential wheel pair and two passive supporting wheels. It is equipped with front and rear laser scanner (Sick LMS). Furthermore, it hosts the power supply and the main part of the computer network.

III. Control Architecture

In this section, we introduce our control architecture. First, we summerize the design criteria. Second, we introduce our designed and hierarchically organized control architecture. Third, we describe the hardware and software modules. The design criteria can be summarized as follows:

- Flexibility and modularity to cope with various tasks and to allow the addition of further tasks and hardware and software modules in a simple manner. This is a very important feature for the process of integration.
- Real-time performance to allow a prompt respond to varying environments and exceptions which can occur during the task execution.

We propose a hierarchically organized control system for our humanoid robot with three levels to handle the complexity of the robot. A given task is decomposed into several subtasks, representing the sequence of actions the subsystems of the humanoid robot must carry out to accomplish the task goal. The coordinated execution of a task requires the scheduling of the subtasks and their synchronization with logical conditions, external and internal events. Figure 2 shows the block diagram of the control architecture with three levels:

- The task planning level specifies the subtasks for the multiple subsystems of the robot. Those could be derived from the task description autonomously or interactively by a human operator. Furthermore, the necessary subsystem controller are selected.
- The task coordination level generates sequen-

tial/parallel primitive actions for the execution level in order to achieve the given task goal. The subtasks are established by the task planning level. The execution of the subtasks in an appropriate schedule can be modified/reorganized by an operator using an interactive user interface.

- The task execution level is characterized by control theory to execute specified sensory-motor control commands. This level uses task specific local models of the environment and objects, which represent the active scene. In the following we refer to those models as *active models*.

The active models are first initialized by the global models and can be modified and enhanced on the basis of the progress of the task execution. Internal system events and execution errors are detected from local sensor data. These events/errors are used as feedback for the task coordination level in order to take appropriate measures. For example, a new alternative execution plan can be generated to react to internal events of the robot subsystems or to environmental stimuli. To achieve a coordinated execution of tasks, a mechanism for the synchronization of actions of the different subsystems of the robot allowing a deterministic switch between sequential/parallel actions was developed [2]. For this purpose, condition-event Petri nets (C/E Petri net) are used to efficiently represent both control and data flow within one formalism.

A. Computer Architecture

The previous computer architecture in ARMAR consisted of an embedded PC and a number of C167 minimodules. These modules were connected to the PC via CAN-Bus. The C167 minimodules acquired the revolution speed of the motor and generated PWM-signals for motor control. As only one CAN-Bus is used in this architecture, the bandwidth of the CAN-Bus has turned out to be a bottleneck for transmission of sensor values and actuator commands. Another problem was the limited cycle time that could be achieved using the C167 minimodule, which is not sufficient for a smooth control of the robot. Recently the computer system in ARMAR was enhanced. At the moment the following components are integrated into ARMAR

- 2 × 1 GHz embedded PC for upperbody motion control, trajectory generation platform navigation, acoustic localisation and speech.
- 1 × 1.7 GHz Laptop for the visual tracking.
- 1 × 700 MHz PC/104 for hybrid position and force control for the dual-arm system.

The current computer architecture consists of several embedded PCs and C167 minimodules (see figure 3). The PCs are connected via ethernet among each other and the connection to the C167 minimodules is established via CAN-Bus sytem. In order overcome bandwidth problems we have distributed the C167 minimodules onto two CAN-Buss.

A new computer architecture [7] was developed to resemble the proposed structure of the control architecture. The computer architecture (see figure 4) is structured into the three following levels:

- task planning level
- synchronization and coordination level
- sensor-actor level

The task planning level is responsible for the scheduling of tasks and management of resources and skills. Complex tasks are divided into subtasks that are transferred to the synchronization level.

The synchronization level coordinates several subsystems of the humanoid robot like arms, head, torso and platform. On this level mainly embedded PCs like PC/104 systems will be used. The interconnection between the

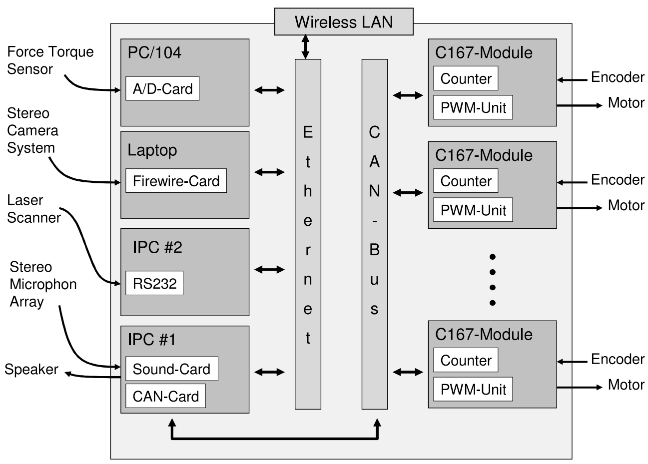

Fig. 3. Hardware architecture in ARMAR

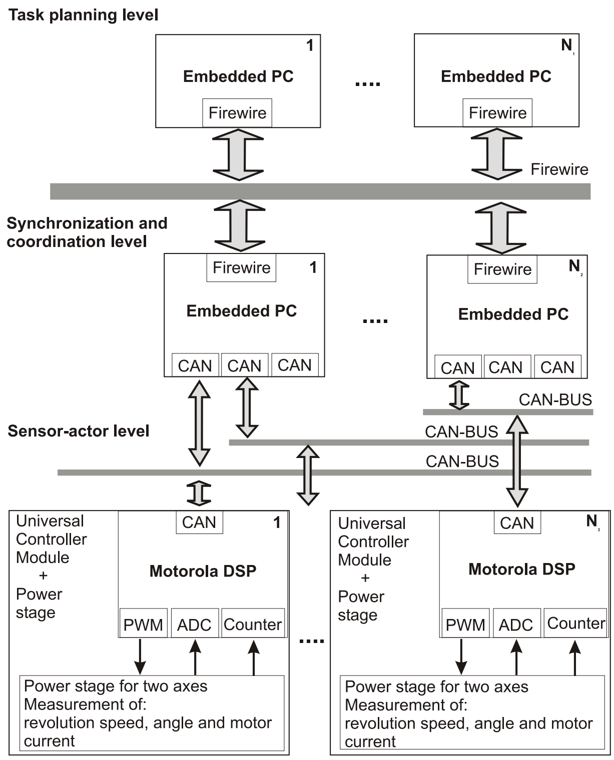

Fig. 4. The new Hardware Architecture

components of the synchronization level will be established by a broadband bus-system like firewire.

At the sensor-actor level the servo control of the motors is done by monitoring the revolution speed of the motor, the angle of the axis and the torque applied to the axis. For this task the Universal Controller Module (UCoM) was developed. The UCoM is a DSP-FPGA-based device which communicates with the embedded PCs via CAN-Bus. By using a combination of a DSP and a FPGA a high flexibility is achieved. The DSP is dedicated for calculations and data processing whereas the FPGA offers the felxiblity and the hardware acceleration for special functionalities. In this case the FPGA's main objective is to take workload from otherwise time-consuming computation on the DSP.

B. Software Environment

- **Operating System:** The computers of the motion control are running under Linux, kernel 2.4.20 with the Real Time Application Interface RTAI 24.1.11. Our on board vision system is running under Windows XP. They communicate over TCP/IP and over sockets with the vision systen.
- **Implementation Framework:** For the implementation of the control architecture we have used the framework MCA version 2.3 ([11], [13]). It provides a standardized module framework with unified interfaces. The modules can be easily connected into groups to form more complex functionality. These modules and groups can be executed under linux, RTAI-Linux and Windows and communicate beyond operating system borders. Moreover, the graphical debugging tool MCAadmin/MCAbrowser which can be connected via TCP/IP to the MCA processes visualizes the connection structure of the modules and groups and provides access to the interfaces at runtime. The MCAGUI provides a graphical user interface with various input and output entities.

IV. Motion Control Methods

In this section we prensent several motion control methods which have been already implemented in order to cope with our task goal, i.e. natural interaction and performing manipulation tasks in a kitchen scenario.

A. Visual person tracking and auditory tracking

In order to make the interaction with the robot easier and more reliable, we integrated the algorithms, which has been originally developed in [3] for the visual perception of the user into the control system of our robot. The 3D-positions of the user's head and hands are mapped into joint angles of the robot head and arms. For the mapping of the hand positions into a robot arm postures, we use the method presented in [1] in order to generate human-like arm postures. Furthermore, an acoustic localisation algorithm using a stereo microphone system, which was developed in [4], is also integrated. Once the head and hands of a person has been detected, the humanoid robot mimics the head and hand motion of the person. When the robot loses the person, the attention of the robot is regained through the acoustic localisation. Our experiments indicate a robust visual and acoustic tracking of a person even when the head is moving. Tracking sessions were running continuously over 2 hours.

B. Arm Motion Control

The execution of single arm manipulation tasks is provided by different inverse kinematics algorithms [1]. This is necessary because most manipulation tasks are specified in terms of the object trajectories. Because of the kinematics redundancy of the 7-DOF-arms of ARMAR, an infinite number of joint angle trajectories leads to the same end-effector trajectory. We use the redundancy to avoid mechanical joint limits and to minimize the reconfiguration of the arm. In Addition, we presented in [1] a method for the generation of kinematically human-like manipulation motions. The method uses a representation for the arm posture suggested in the neurophysiology in order to determine the joint angles of the shoulder and elbow θ_1, θ_2, θ_3, and θ_4 of the robot arm, which specify the position of the wrist. The remaining joints angles θ_5, θ_6 and θ_7 are used to determine the exact position and orientation of the end-effector.

C. Self-Collision Avoidance

The detecting and avoiding self-collisions is essential in the case of multiple arm robots, which has to safley operate in household environments. For the self-collision avoidance of the dual-arm system of our humanoid robot, we have implemented a simple 3D-collision avoidance method. For this purpose, we modeled all upperbody parts of the robot (arms, hands and torso) by bounding cylinders terminated with semispheres. The setup of the bounding cylinders have been created as follows: Each arm consists of three cylinders where the one covers the hand, one the lower arm

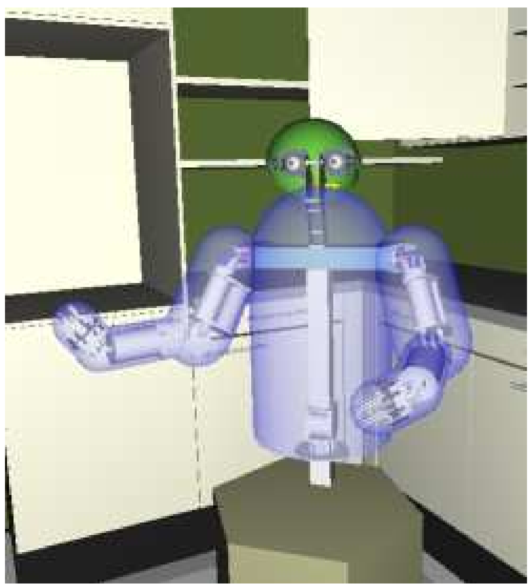

Fig. 5. Collision hulls of Armar: Each arm consists of three cylinders whereas the torso is modeled as a separate cylinder

and one the upper arm. The Torso is modeled as a separate cylinder (see figure 5).

Due to the mechanical joint limits only 26 out of 36 possible collisions have to be verified. On an AMD Athlon 1GHz we have reached an average computation time of 2.2 milliseconds for the upper body. Unlike [8] where the convex hull of polyhedra are used, our approach with simpler geometric shapes does not require a very high demand of computing power. Furthermore, the method can be applied to collision avoidance between the robot and external obstacles.

D. Locomotion Control

The locomotion control is designed to provide multiple driving strategies under different situations. On the one hand we want the platform to find its way from one location to another within a household environment and to avoid collisions autonomously. On the other hand, the control over the motion path and speed needs to be passed on to other entities of the robot such as the module for the coordination of the whole body motion.

For satisfying the requirements mentioned above, the platform basically needs to know where it is. This is the task of the navigation done by an odometry and a pose correction. Based on the pose information a path computation can determine a path from one place to another. Then the planned path can be observed and checked by the path observation. It is responsible for setting a new target if stopovers have been reached on the path. The path segment to these target has to be observed by the collision avoidance. It does a local replanning of the path when obstacles occurr on the segment ahead. Finally, a spline driving module controls the execution of the motion on the path by generating desired values for a kinematics module which transforms the control values to wheel speeds for the motor controllers. Beside these autonomous control modules, a manual control for the purpose of placing and maintaining the platform has been considered very useful.

According to this idea the overall structure which we have implemented for the control of the platform currently consists of six major modules:

- manual driving
- navigation by odometry and scanner based pose correction
- graph module
- collision avoidance
- spline path driving
- kinemtics
- motor control

In the remainder of this section a more detailed description of the above mentioned modules is given.

We begin with the directly user controlled branch. The manual driving provides manual control of the platform with a joystick for the ease of handling when the robot is not running autonomously and needs to be moved. For this purpose the joystick's x and y-axis values are mapped to radius and speed respectively for the kinematics module.

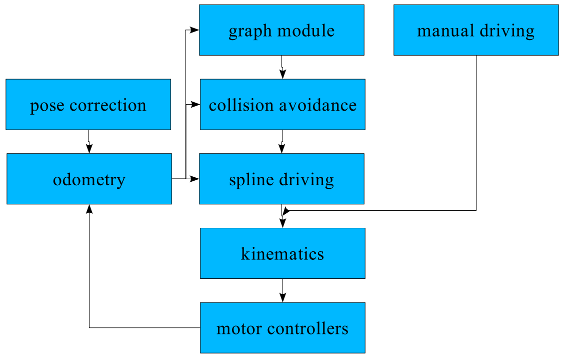

Fig. 6. Architecture of the platform control

On the autonomous branch the fundamental modules are the odometry and the pose correction. They are working closely together. Due to slip and limited precision the odometry would lose the vehicle's pose after some time as the measuring errors would accumulate to a very high extent. For this purpose the scanner distance data is being used in conjunction with a map in order to calculate corrections. We have implemented a modified method for matching a local line segment map to a global line segment map (see Fig.7) based on [10]. As we are using laser scanner range data with far less uncertainty and higher precision than the ultrasonic sensors originally used in [10] the matching of the extracted edges is more precise.

We have measured a precision of 2cm in the best cases and 4cm in the worst case. Still improvement is needed as the matching process may fail when there are too many obstructions such as chairs, other furnitures or persons which are mobile and therefore not registered in the map.

Based on a topological graph the graph module computes paths consisting of a chain of nodes which delivers stopover positions in the motion plane from an actual to a target node. The search algorithm is the well known A* algorithm. In order to be more flexible the nodes contain a boundary description of the area encompassing

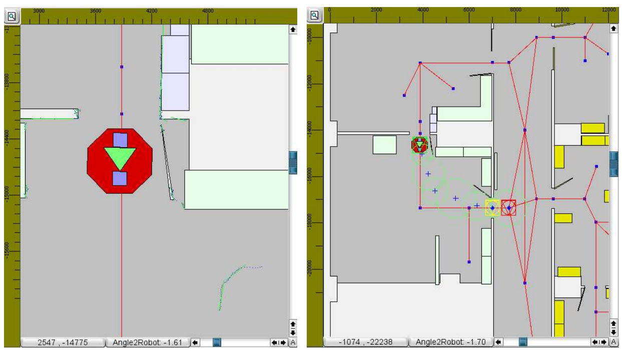

Fig. 7. Repositioning with extracted edges and collision avoidance with elastic bands

a node. This additional information allows a permanent reassignment of the current position to the closest node in the graph. By this fact the mobile base is enabled to interact with the user and the events in the environment arbitrarily and retrieve its current node in the graph in order to be able to find a way to distant destinations whenever needed. For a better handling of changing situations in respect of long term obstructions and door transitions a diversification of node types is made. We differ between room nodes containing the above mentioned boundary description, location nodes which contain rectangles for describing a smaller long-term free space area and door nodes containing additionally the coordinates of the door axis with the turn direction for opening. This information supplement enables the upper level to prepare and act accordingly, should the door be closed at this node. The detection of open or closed state can be easily done by simple analysis of the distance image of the laser range scanner when the platform approaches the coordinates of the door axis. Additional edge attributes like motion speed allow an adjustment to the potential threats of collision at certain parts of the virtual tracks between the nodes such as doors in a corridor. For enhancing the versatility of the graph we intend to make it modifyable at run-time. This will allow the cognitive level to adapt it to changing conditions and extend it with new nodes and informations.

The path segment passed on from the graph module is being observed by the collision avoidance by the analysis of the scans. If obstacles occur the path will be modified according to the elastic band approach from [9] in order to avoid collision (see Fig.7).

If the path section is fully obstructed the collision avoidance reports this to the graph module for temporary disabling the current edge and replanning on the graph level. In the case of a graph unrelated motion the collision avoidance receives a number of support points from the upper levels bypassing the graph module.

Finally, the spline driving executes the driving on this potentially modified path by calculating the required speed and radius passing them on to the platform kinematic module which transforms them into the desired speeds of the motor controller. For specific movements like i.e. door opening the driving path can be set by upper levels bypassing both the graph module as well as the collision avoidance. Yet, the motion speed can be dynamically adjusted in order to synchronize the platform motion with the upper body motion.

V. Conclusion and Further Work

In this paper we presented the current state of our work in developing and integrating motor and perception components for humanoid robots. We prestented a modular and distributed control architecture which allow the integration of further hardware and software components.

Further work includes the integration of five-fingered lightweight hand [5] as well as a speech recognition system. In addition, methods for local path modification in order to achieve a coordinated motion of the whole humanoid robot (platform, torso, arms and head) are addressed in the near future. For this purpose, multisensor based methods will be investigated and implemented to ensure a collision free motion of the whole robot in the kitchen.

Acknowledgment

This work has been performed in the framework of the german humanoid robotics program SFB 588 funded by the German Research Foundation (*DFG: Deutsche Forschungsgemeinschaft*).

References

[1] T. Asfour and R. Dillmann. Human-like Motion of a Humanoid Robot Arm Based on Closed-Form Solution of the Inverse Kinematics Problem. The IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2003), Las Vegas, USA, 2003.

[2] T. Asfour, D.N. Ly, K. Regenstein and R. Dillmann. Coordinated Task Execution for Humanoid Robots. The 9th International Symposium on Experimental Robotics (ISER 04), Singapore, 18-21 June 2004.

[3] K. Nickel and R. Stiefelhagen. Pointing gesture recognition based on 3d-tracking of face, hands and head orientation. International Conference on Multimodal Interfaces, Vancouver, Canada, 2003.

[4] D. Bechler, M. Schlosser and K.Kroschel. Acoustic 3D Speaker Tracking for Humanoid Robots with a Microphone Arry. Proceedings of the 3rd IEEE International Conference on Humanoid Robots (Humanoids 2003), Karlsruhe, Germany.

[5] S. Schulz and G. Bretthauer. A Fluidic Humanoid Robot Hand. Proceedings of the second IEEE International Conference on Humanoid Robots (Humanoids 2001), Tokyo, Japan.

[6] R. David and H. Alla. Petri Nets for Modeling of Dynamic Systems, Automatica, vol. 30, pp. 175–202, 1994.

[7] K. Regenstein and R. Dillmann. Design of an open hardware architecture for the humanoid robot ARMAR. Humanoids 2003, International Conference on Humanoid Robots, Karlsruhe, Germany, 2003.

[8] J. J. Kuffner, K. Nishiwaki, S. Kagami, Y. Kuniyoshi, M. Inaba and H. Inoue. Self-Collision Detection and Prevention for Humanoid Robots, In Proc. 2002 IEEE International Conference on Robotics and Automation (ICRA 2002), 2002

[9] S. Quinlan and O. Khatib, Elastic Bands: Connecting Path Planning and Control, In Proc. 2002 IEEE International Conference on Robotics and Automation (ICRA 1993), 1993

[10] J. Crowley. World Modeling and Position Estimation for a Mobile Robot Using Ultrasonic Ranging. In Proceedings of IEEE International Conference on Robotics and Automation, Scottsdale, AZ, May 14-19, pp. 674-680, 1989.

[11] K.-U. Scholl , J. Albiez and B. Gassmann. MCA - An Expandable Modular Controller Architecture, 3rd Real-Time Linux Workshop, Milano, Italy, 2001

[12] ATI Industrial Automation Homepage: http://www.ati-ia.com/sensors.htm

[13] Modular Controller Architecture (MCA) http://mca2.sourceforge.net/

[14] Sick-Homepage: http://www.sick.de/de/products/