FUmanoid
Team Description 2007

Hamid Reza Moballegh, Roman Guilbourd, Gretta Hohl, Arturo Reuschenbach, Jinping Yu, Raúl Rojas

Institut für Informatik, Arbeitsgruppe Künstliche Intelligenz,
Freie Universität Berlin, Takustr.9, 14195 Berlin, Germany
http://fumanoid.mi.fu-berlin.de

Abstract. This document describes hardware and software of the robots developed by the "FUmanoid" Team for the RoboCup competitions to be held in Atlanta 2007. The mechanical structure of the robots is based on the Bioloid construction kit, which is available for research and competition. The platform has been then customized and equipped with additional hardware such as a camera and a PC104+ single board computer module, on which the control programs run. The original version of the robot has also a microcontroller which is still being used for low level control. This paper explains our control and stability methods as well as the software and hardware used for the robot.

1. Introduction

The FUmanoid team was started in 2006 in the Artificial Intelligence group at Freie Universität Berlin, which has had a successful and long history in RoboCup with the FU-Fighters team. Many of the hardware and software techniques developed for the FU-Fighters are now available to be used in several other projects, such as our humanoid soccer robots. Humanoid robots have many potential applications, which make this area very attractive for researchers. The FUmanoid project is a step towards research and development of robots which offer more real human-interaction, can perform tasks in our environment and will be able to play important roles in our daily life.

2. Hardware Design

2.1 Mechanical Design

The Bioloid Robot Construction Kit produced by Robotis Inc. has been used as the initial mechanical platform for our FUmanoid robots. The original robot kit is based on an 18 DOF motion mechanism with six actuators per leg and three per arm. Because of the modular mechanical structure of the system, it is easy and straightforward to modify the system, or even add or reduce some of the degrees of freedom. This is of great importance because the region of stability of the motion system is strongly dependent on the mechanical construction.
A modification made to the original mechanism, is adding the required PC104+ module, a camera and batteries to the system. The upper trunk is being used as a place to hold the PC104+ module. This module is a bit heavy and makes the stabilization of the robot somewhat harder, but it has the advantage of having more processing power and the availability of a variety of accessories, such as cameras. It is also easier to write the software over standard IDEs like

Microsoft visual C++, or even use external PCs in the development phase. In addition, the weight distribution is more human-like, having this module installed.
A fixed camera has been attached to the body. Its motion can be obtained from the other degrees of freedom of the robot; i.e. the robot turns the upper trunk or the whole body when it wants to search for the ball. Fig.1 shows a photograph of the robot together with its dimensions.

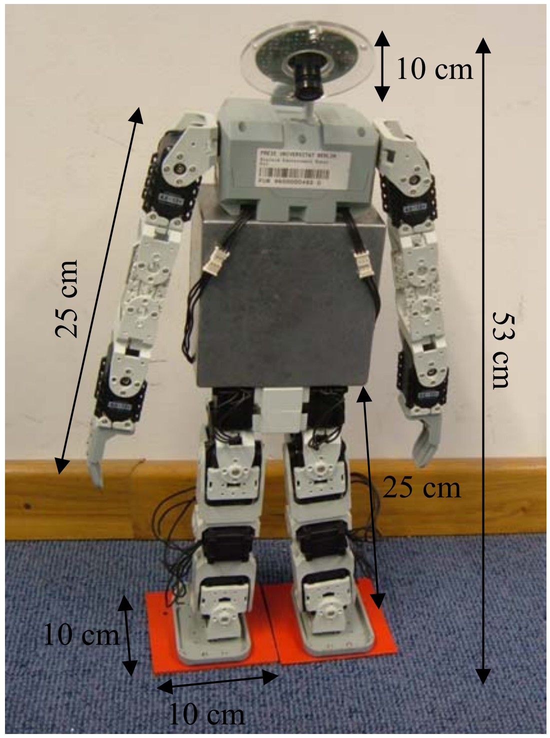

Fig.1: Robot hardware

2.2 Actuators

The actuators used in our robot are “Dynamixel AX-12” servomotors, produced by Robotis Inc. Each actuator has its own microcontroller which implements adjustable position control using potentiometer position feedback. It also calculates many other parameters such as rotation speed and motor load which can be accessed through a single-bus, high-speed serial communication protocol. This facilitates the construction of an extendable network of motors which can be individually accessed and controlled by a single microprocessor. The original kit is also equipped with an ATMEL AVR based central processing unit, which can be programmed either with a simple (but not so efficient) development environment or directly in C.
Some of the parameters of the actuators are the following:
– Weight: 55 g
– Gear Reduction Ratio: 1/254
– Max Holding Torque: 16.5 kgf.cm (@10V)
– Speed: 0.196 sec/60 degrees (@10V)
– Resolution: 0.35 degrees

2.3 Sensors

Currently (20 Jan. 2007), the robot has only two types of sensors: the camera and the sensory feedback from the servomotors. The goal of the developer team is to use a minimum-number of sensors to avoid complexity and minimize cost, together with a focus on software based control and stabilization methods.

The camera type used in FUmanoid robots is an IEEE1394 webcam, which delivers color frames (15 fps, 640x480 pixels, using YUV422 format). The camera and its interface are compatible with the ones used by the FU-Fighters. Therefore the already developed image acquisition modules have been used in image processing software.

The sensory feedback of the actuators includes the current joint angle, the current motor speed, and the load. Because all of this values are derived from the only feedback sensor of the actuators (the position potentiometer), the latter two values are not so reliable. There are also other measured values which can be accessed through the Dynamixel serial interface, such as supplied voltage and temperature, which can be used for safety purposes.

2.4 Processors and communications

Since the original processing unit of the robot (called CM5) is very light and many of the functions needed to communicate with actuators and PC are available for this platform, this unit was not removed from the system but was reprogrammed using C, and used as low-level controller of the robot. The main Processor of the CM5 is an ATMEL ATMEGA128 which is an 8-bit RISC microcontroller clocked at 16MHz. This microcontroller has plenty of resources, among them 2 USART modules which are used to communicate with both sides. In the servo side 1Mbps is used as the baud rate. Each servomotor has a unique ID for packet identification. There is also a broadcasting ID used to send the same data packet to all existing motors on the bus. On the PC side the communication is performed using a standard 57600bps RS232 protocol.

The main processing unit is a PC104+ biscuit module with a 600MHz Intel Pentium Celeron (Fig. 2.). The module is also equipped with an IEEE1394 extension board, which provides the required ports for connecting the camera. Other peripherals such as wireless network modules will be attached to the system using USB ports.

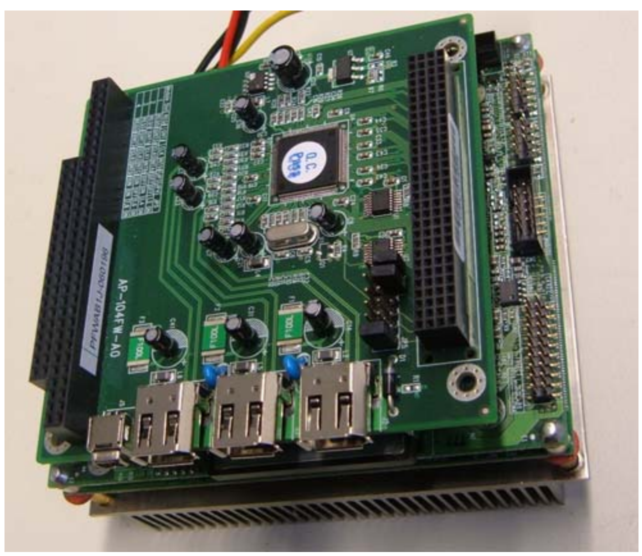

Fig. 2: The main processor built on the robot

3. Software Design

Fig.3 shows the block diagram of the software which runs in the robot's main computer. The program consists of 4 sub-blocks:

- Hardware interface: The only part of the control program directly connected to the hardware. This part is responsible for all of the hardware related tasks, i.e. power management and accessing actuators and sensors.
- Network interface: Mainly responsible for the wireless communication of the robot with the other robots or the referee box.
- Vision engine: contains all image processing algorithms, acquires images from the camera, and calls different recognition functions for calculating and updating vision-based sensory data for the rest of the program.
- AI engine: The behaviors of the robot are programmed in the AI engine. Each behavior can access the data provided by different sources and send proper commands to the actuators through the hardware interface.

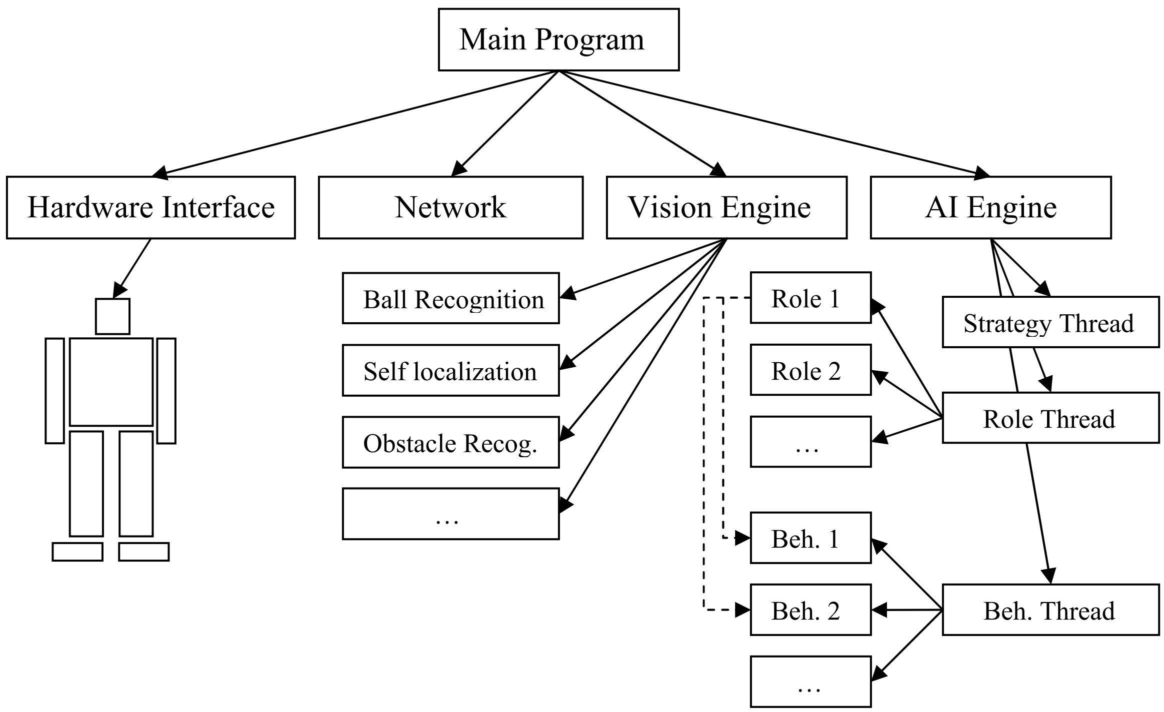

Fig. 3: Structure of the control software

3.1 control and stabilization

Stabilizing humanoid robots is a challenging subject which has attracted many researchers who have developed widely varying techniques. These techniques range from simple and static COG methods to poorly dynamic nonlinear control methods using multi-DOF underactuated inverted pendulum models. A novel approach was developed for the biped walking stability problem by McGeer, who pioneered the idea of passive dynamic walking. This approach which is both simple and direct has been followed by Collins, Wisse and Ruina and has been improved with different techniques to obtain 3D walking stability.

Our approach is to follow the same idea during walking and improve it to have more control over the robot to change the walking direction and make the robot stop or start walking, mixed with other mostly active behaviors such as kicking and standing up. Towards this goal, we simulate the robot with ODE and its 3D walking stability has been tested in simulations. But implementing the idea in the physical robot seems to be difficult because it is not possible to make the needed joints, move frictionless or apply the exact desired torque to them without modifying the current actuator hardware. There are two main problems. First, the gear reduction ratio acts as a strong damper with great energy loss, although energy of

center of mass is of great importance in passive dynamic walking. Second, it seems that the Robotis motors act as if their terminals are shorted to each other when they are switched off by the software. This makes the energy loss even worse.
As a test, the friction of the ankle actuators of each foot was reduced by removing a gear from each, so that they could only function as low friction joints with position sensors, the data derived from this sensors are then used in the control program which finds the stance foot at each step and controls all active actuators regarding to the stance angle. The robot has been able to walk several steps on a slope. To make the robot walk on a flat surface and perform other tasks needed in RoboCup humanoid league, it was required to actuate the disabled joints simultaneously. One possible solution to this problem can be rewriting the firmware of the actuators, a task currently in progress.
Because several other necessary behaviors of the robot are fully active and mostly semi-static, a key-frame generation program was developed and is used to generate stable trajectories for different behaviors. Other dynamic behaviors like walking, rotating and kicking have also been developed successfully but a bit slow with this method to make the development of the other parts of the software independent from the progress of our passive dynamic research.

3.2 Vision

The vision in FUmanoid robots is able to recognize objects such as the ball, goals, corner poles and other robots/obstacles in the field. Object recognition procedure uses a color based region growing method. In order to localize the robot, a particle filter has been implemented, which uses several features detected in each frame together with some results from other robot(s) to enhance the position and orientation hypotheses.
To calibrate and mark the existing color ranges in the color space, a color calibration program has been developed, which enables the user to define custom 3D geometries in this space. The defined geometries surround each point cloud and can be edited and generalized with different lighting conditions. Fig. 4 shows a snapshot of the color calibration program. Each of the geometries has been defined as a 3D multi section pipe with an adjustable diameter at each section. The next phase is to make the parameter calculation of these geometries adaptive and automatic.

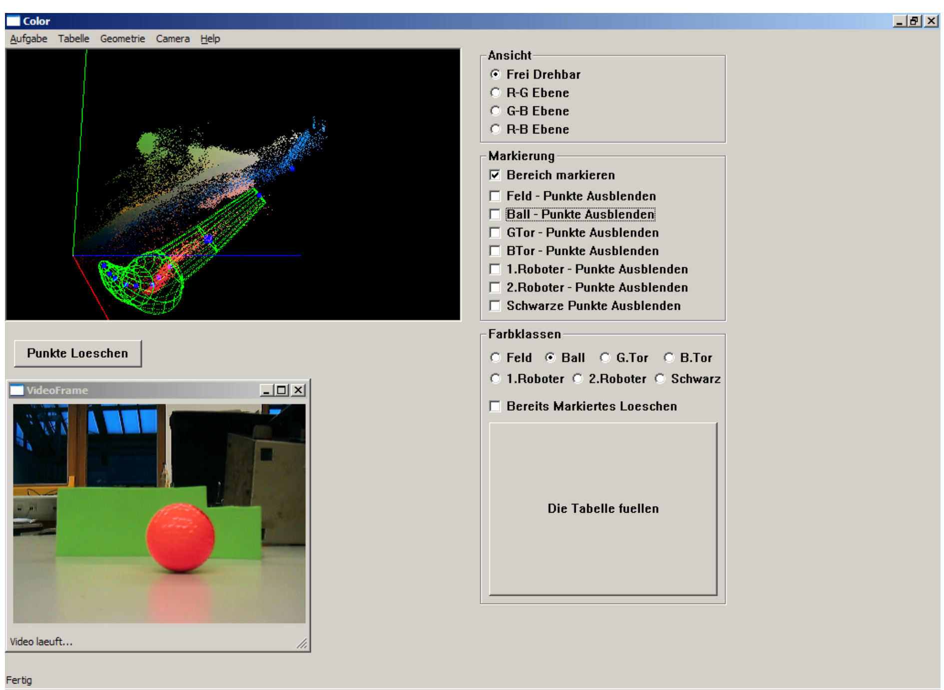

Fig. 4: a snapshot of the color calibration program

3.3 Decision Making and Robot Behaviors

For the construction of the FUmanoid robots, a new method of decision making is used, which facilitates the generation of complicated multilayer behaviors. In this method several threads run concurrently, each of which contains a scenario. In our FUmanoid robots there are scenarios at three levels, named Strategy, Role and Behavior. The advantage of using scenarios instead of other decision making methods such as decision threes or the FSM concept is that each scenario runs continuously (is not called periodically) until the behavior is finished. This not only avoids inconsistencies and unwanted loops in the decisions, but also makes the written source code very understandable, because it is being written what is expected to be done by the robot. The behavior layer is mainly responsible for small behaviors, like walking, rotation, etc. Most of these programs are very hardware dependent and can only be developed trough direct experiment with the hardware. The upper layer contains scenarios named roles, which are more general and less dependent on hardware. Roles combine several behaviors to make a real player. Some examples are attacker, defender and goalie. The Strategy layer is responsible for the whole approach of the team, which can also be placed over an external server. Cooperative tasks can also be managed in this layer.
Although this is too much for managing only two robots in the current structure of the RoboCup humanoid games, it can help a lot in future as the number of robots and complexity of the game will increase. The active scenario in each layer can be changed from the upper layer but often related to the status feedback of the current running scenario.

References

1. Felix von Hundelshausen, Michael Schreiber, Fabian Wiesel, Achim Liers and Ra´ul Rojas: *MATRIX: A force field pattern matching method for mobile robots*, Technical Report B-08-03, Freie Universität Berlin, Institute of Computer Science, Takustr. 9, 14195 Berlin, Germany, available at: http://robocup.mi.fuberlin.de/docs/matrix.pdf
2. Felix von Hundelshausen, and Raúl Rojas: *Tracking Regions*, in Daniel Polani et al.(editors):RoboCup-2003: Robot Soccer World Cup VII (Lecture Notes in Computer Science), Springer, 2004.
3. Felix von Hundelshausen, *A constructive feature-detection approach for mobile robotics*. Proceedings of the RoboCup 2004 International Symposium, Lisboan, Italy July 5-7, 2004.
4. *A Three-Dimensional Passive-Dynamic Walking Robot with Two Legs and Knees,* Collins, S. H., Wisse, M., Ruina, A. International Journal of Robotics Research, Vol. 20, No. 2, Pages 607-615, 2001
5. McGeer, T. 1990. *Passive dynamic walking*. International Journal of Robotics Research 9(2):62–82.
6. Erik V. Cuevas, Daniel Zaldívar, and Raúl Rojas: *Robust fuzzy segmentation*, in International Congress on Computer Sciences, Biomedical Engineering and Electronics CONCIBE 2005, Guadalajara, Mexico, October 2005
7. Erik V. Cuevas, Daniel Zaldívar, and Raúl Rojas: *Vision tracking prediction*, in International Congress on Computer Sciences, Biomedical Engineering and Electronics CONCIBE 2005, Guadalajara, Mexico, October 2005
8. Erik V. Cuevas, Daniel Zaldívar, and Raúl Rojas: *Incremental fuzzy control for a biped robot balance*, in International Conference on Robotics and Applications 2005, October 31 - November 2, 2005 Cambridge, USA
9. Eric V. Cuevas, Daniel Zaldívar, and Raúl Rojas: *Dynamic Control Algorithm for a Biped Robot*, in 7th International Conference on Control and Applications, Cancún, Mexico, May 18-20, 2005.