The ZeeRO Mobile Robot - A Modular Architecture

Gheorghe Lazea[1],Radu Bogdan Rusu[1,2],Radu Robotin[1],Romulus Sime

[1]Robotics Research Group, Department of Automation
Technical University of Cluj-Napoca, C. Daicoviciu 15, Cluj-Napoca, Romania
E-mail: *{gheorghe.lazea; radu.rusu; radu.robotin}@aut.utcluj.ro*
[2]Intelligent Autonomous Systems, Computer Science Department
Technische Universität München, Boltzmannstr. 3, Munich, Germany
E-mail: *rusu@cs.tum.edu*

Abstract: *This paper presents the general architecture of the ZeeRO mobile robot, with an emphasis on its main modules: navigation, sensors and data processing. The interface with the Player server that allows simulations in 2D (Stage) or 3D (Gazebo) is also presented. The last part of the paper presents some applications for this mobile robot, some of the performed tests and also some of the future plans for the extension of the current architecture.*

Keywords: *mobile robot architecture, Player server, Gumstix, sensor data processing*

I INTRODUCTION

The "ZeeRO" mobile robot was developed by the Robotics Research Group in the Automation Department, Technical University of Cluj-Napoca. The aim of the project was to design a low-cost, modular structure (Open System Architecture) mobile robot, that allows further extensions. The robot was designed to be embedded in individual applications (i.e. static or dynamic environment navigation), but also in a more comprehensive system, easy to integrate with existing robots (Pioneer 2 and Pioneer 3) in cooperative applications.

The system uses the Player/Stage platform [3], and integrates new high level functions and algorithms, thus making the mobile robot programmable in a variety of programming languages (Java, C/C++, Lisp, Python, etc). Figure 1 shows the mobile robot architecture. The central processing unit uses a distributed architecture with a distinct separation between high level functions (master) and low level operations (slave).

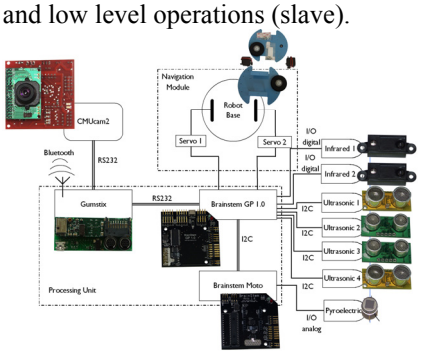

Figure 1. The ZeeRO architecture

The link with other mobile robots, but also the communication between the ZeeRO mobile robot and a PC is wireless, therefore providing a higher degree of autonomy and flexibility to the overall system. The design aim for the sensors module was a reliable data acquisition system that could facilitate:

obstacle detection and avoidance, color spot detection, human (or any warm object) detection and following, as well as support for map building and collision avoidance.

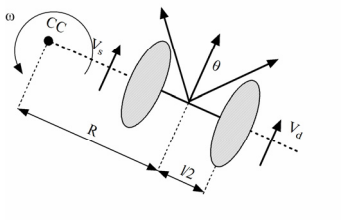

Figure 2. Motion robot parameters

II THE NAVIGATION MODULE

The ZeeRO mobile robot is a two wheels differential drive vehicle. The two servomotors are directly controlled by a component of the central processing unit. Two additional servomotors can be found in the CMUcam2's pan-tilt unit. They are directly controlled by the SX52 microcontroller (up to 5 servomotors can be controlled). The robot motion control parameters are x and y, that is the position of the center of the wheel axis, and θ, that is the orientation of the mobile robot with respect to the horizontal X axis (see figure 2). These parameters are determined as a function of time, using:

$$\begin{cases} x(t) = \frac{1}{2}\int_0^t \left[v_d(t) + v_s(t)\right] \cdot \cos\left(\theta(t)\right) \cdot dt \\ y(t) = \frac{1}{2}\int_0^t \left[v_d(t) + v_s(t)\right] \cdot \sin\left(\theta(t)\right) \cdot dt \\ \theta(t) = \frac{1}{l}\int_0^t \left[v_d(t) - v_s(t)\right] \cdot dt \end{cases} \quad (1)$$

The set $[x(t), y(t), \theta(t)]$ determines the position and the orientation of the mobile robot at time t, starting at the initial conditions $[x_0, y_0, \theta_0]$. The equation (1) was determined based on the forward kinematics of the robot, that is:

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = [R] \begin{bmatrix} x - x_c \\ y - y_c \\ \theta \end{bmatrix} + \begin{bmatrix} x_c \\ y_c \\ \omega \cdot \partial t \end{bmatrix} \quad (2)$$

where R is the rotational matrix, given by:

$$R = \begin{bmatrix} \cos(\omega \cdot \partial t) & -\sin(\omega \cdot \partial t) & 0 \\ \sin(\omega \cdot \partial t) & \cos(\omega \cdot \partial t) & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (3)$$

The last equation describes the rotation of the mobile robot around point C (center of curvature - see figure 2) with the angular velocity, ω, while v_d and v_s are the velocities of the right and left wheel, given by:

$$\begin{cases} v_d = \omega \cdot \left(R + \frac{l}{2}\right) \\ v_s = \omega \cdot \left(R - \frac{l}{2}\right) \end{cases} \quad (4)$$

where R is the curvature radius and l is the distance between wheels.

Two special cases of motion are encountered:

- Forward motion, the orientation for the start configuration is the same as the orientation for the target configuration ($\theta_s = \theta_t$) and the velocities of the two wheels are identical ($v_d = v_s = v$)
- The robot turns around point O (the robot turns in place), $v_d = -v_s$.

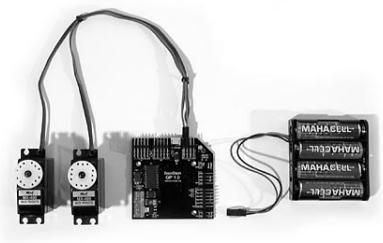

Figure 3. Brainstem GP1.0 board and servomotors

The locomotion module uses two servomotors, modified for continuous motion (see figure 3 - Modified Servomotor – 111 OZ - In). The motors are PWM controlled, with a period of 20ms, the rotation angle being determined by the pulse width. For example, a pulse width of 1.5 ms determines a 90 degrees rotation in the motor shaft (see figure 4). The servomotors were transformed into continuous motion motors (see figure 5) by replacing the feedback sensor with a resistor array, that will transmit a signal to the motor similar to the one sent by the feedback sensor when the motor shaft is at 90 degrees.

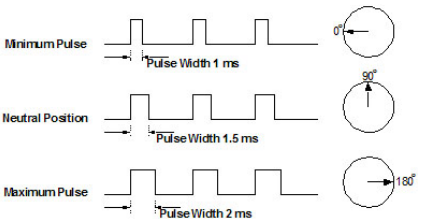

Figure 4
Rotation angle function of the command pulse width

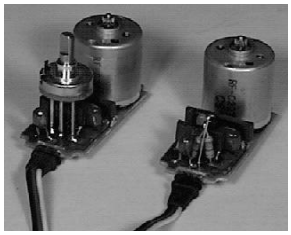

Figure 5. Servomotor with feedback sensor (left) and with resistor array (right)

As a result, if a signal for 0 degrees motor positioning is applied, the motor will rotate with maximum speed in one direction, or, if a signal for 180 degrees motor positioning is applied, the motor will rotate in the reverse direction with maximum speed. Thus, by removing the feedback sensor the motor behaves as if the shaft would be at 90 degrees and will continuously rotate in one direction as long as the command signal persists.

The motors are controlled using Brainstem GP1.0 (see figure 3), built around a 40MHz RISC processor. The Brainstem GP1.0 module uses an 8-bit value to control the position of the servomotor and allows fine tuning of the parameters for the PWM command signal at each sample. A command value of 128 will stop both motors, while a command value of 0 or 255 will rotate the motors with maximum speed in one direction (0 for left and 255 for right). This is achieved by using configuration commands for the output of the Brainstem GP1.0 board. The programming of the two modified servomotors includes: configuration (through software), motor programming (using TEA – Tiny Embedded Application, Assembler or C) and trajectory control routines.

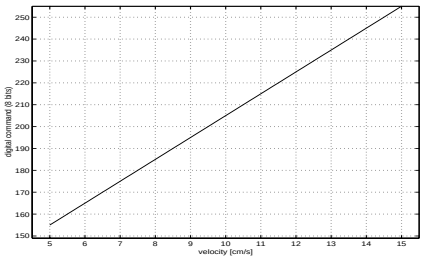

Figure 6
Command signal function of desired linear velocity

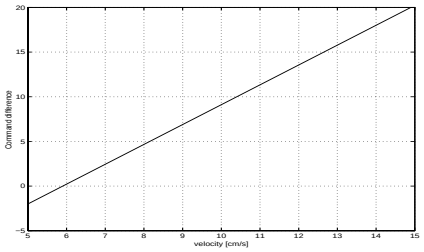

Figure 7
Difference in command signals in function of the desired linear velocity

The two motors are not synchronized at the same command signal. This means that the motors will have different velocities at the same command signal and, therefore the robot will have a trajectory deviation. A relation between the command signal and the velocity has to be found. Figure 6 and figure 7 show experimental results for one direction..

III THE SENSORS MODULE

The role of the sensors module is to acquire data for various navigation controlling algorithms (see figure 8). The following type of sensors are part of the ZeeRO mobile robot's sensorial system:

- 2 IR sensors (Sharp GP2DO2) with an operating range between 10 and 80cm;
- 4 Ultrasonic sensors (2 Devantech SRF08 and 2 SRF10) with an operating range between 3cm and 6m and an accuracy of ±3cm;
- 1 Pyro-electrical sensor (Acroname ELTEC 442-3);
- 1 CMUcam2 video camera (CCD OV6620 sensor and SX52 processing board) connected via RS232 with the main processing unit.

More in-depth information about the sensorial system can be found in [1], and [4].

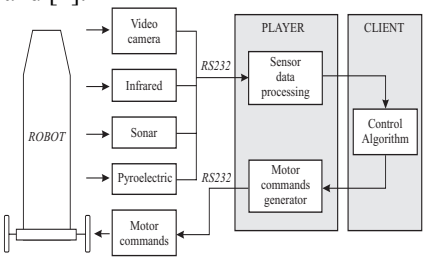

Figure 8. Client and PLAYER module diagram

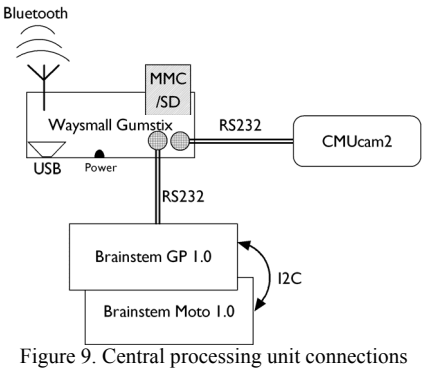

Figure 9. Central processing unit connections

IV THE PROCESSING UNIT

Figure 9 shows the connections between the main elements of the central processing unit. The Gumstix Waysmall provides the high level layer while the Brainstem GP1.0 and Brainstem Moto1.0 represent the low level layer.

The Gumstix Waysmall is an embedded device, based on Intel's

PXA255 XScale processor, operating at 400 MHz, with 64 Mb RAM and 4 Mb Flash memory. Gumstix is powered by the Linux operating system. The Brainstem GP1.0 is built around a 40MHz RISC processor. It provides 5 10bit analog/digital input channels, 5 digital I/O lines, an I2C interface, 4 servomotor outputs, 1 IR port, and a RS232 serial port. The Brainstem Moto1.0 is also built around a 40MHz RISC processor. It provides 2 PWM controlled channels, 1 analog /digital (10bit) input channel, 1 digital I/O channel, an I2C interface, a RS232 serial port, and a H-bridge connector.

For the ZeeRO mobile robot, the Brainstem network comprises one GP module and one Moto module (see figure 9), stacked together via I2C (up to 126 modules can be connected). This approach provides several advantages:

- support for communication between the modules;
- only one serial connector is used to interface the network with the PC;
- only one power supply is used for the network, the Moto module being supplied via the GP module.

V THE PLAYER/STAGE ARCHITECTURE

The development of the Player/Stage project was started at the University of Southern California, USA (1998) [3], and later (2001) moved to SourceForge. Player/Stage has two main components: the server component (Player), which provides an interface for the mobile robot (sensors and actuators) and the simulator component (initially Stage, for 2D simulations, and later Gazebo for 3D).

The Player/Stage system runs on a variety of operating systems: Linux, MacOS, *BSD, etc. Figure 10 presents the global architecture using the server (Player) and client applications (physical devices and 2D and 3D simulators).

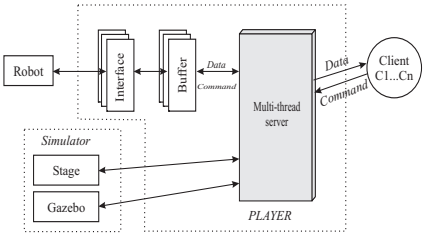

Figure 10. The Player server architecture

Player uses a client-server model based on the TCP (or UDP) protocol, thus enabling the client applications to be implemented using almost any programming language. Furthermore, the client may run on any computer that is connected to a network where the Player server runs.

The developers of the Player project [3] have made a clear separation between the "programming interface" and the control structure. The Player server is implemented in C++ using standard POSIX *pthread* functions to provide support for multithreading. Each client uses a TCP/UDP socket in order to connect to the Player server. If both the client and server are on the same computer, the connection will be a loop-back. On the other hand, Player connects to physical devices, most of the time using RS232 (see figure 8). Numerous enhancements have been added to the Player/Stage project since its initial release on SourceForge. The last version (currently, 2.0.1) supports a wide variety of sensors and actuators, as well as well-known

mobile robots. The community has grown rather large and the code repository is being supported by an international community of robotics researchers.

As figure 10 shows, Player receives information from the mobile robot (which can be a real mobile robot or a simulated one) usually through the serial port. This data is processed by the Player driver and then it is made available to the client. The approach is flexible, new devices may be added, either as a module in the Player server or as a dynamic link library that Player will load before running the client.

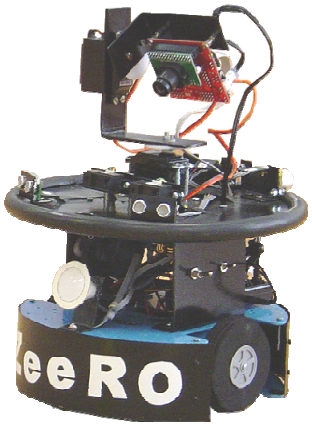

Figure 11
The ZeeRO mobile robot

Results, conclusion and further work

The ZeeRO mobile robot (figure 11) is the result of this research project, still under development. The mobile robot was tested under several conditions, including: obstacle detection, human detection (or heat detection using the pyro-electrical sensor), detection and follow of a color blob (using the CCD sensor), navigation in static and dynamic real environments. Navigation applications in simulated environments (Stage and Gazebo) where also developed. Currently, we are researching the use of cooperative applications for multi-robot systems (using Robotux[2]).

Further work on this project will be focused on extending the sensorial system, increasing the performances of the locomotion module and the sensor data fusion algorithms.

References

[1] Radu Bogdan Rusu. Modern architectures for mobile robots: Javaclient and ZeeRO. *Dissertation thesis for Advanced Postgraduate Studies*, June 2005.

[2] Radu Bogdan Rusu, Liviu Miclea, and Szilard Enyedi. Robotux – a multi-agent robot based security system. In *Proceedings of IEEE-TTTC Automation, Quality & Testing, Robotics International Conference 2004, Cluj-Napoca, Romania, May 13-15, 2004, AQTRJ.*

[3] B.P. Gerkey, R.T. Vaughan, A. Howard, - The Player/Stage project: Tools for Multi-robot and Distributed Sensor Systems, in Proceedings of the International Conference on Advanced Robotics (ICAR), Coimbra, pp 317 – 323, 2003.

[4] Sime Remus. A sensorial system for the ZeeRO mobile robot. *Diploma thesis*, June 2005.

[5] Radu Robotin and Gheorghe Lazea. Path planning in an unknown environment. In *International Conference A&QT-R, 23 May 2002, Cluj-Napoca, Romania, vol.II, Pag.83.*