A new robotic system based on reusable robotic modules and a robotic middleware

Daniele Caltabiano[1], Davide Ghezzi[1], Roberto Sannino[1],
Luca Spelgatti[1] and Davide Brugali[2]

[1]STMicroelectronics {daniele.caltabiano, davide.ghezzi,
roberto.sannino, luca.spelgatti}@st.com
[2]Università degli Studi di Bergamo (brugali@unibg.it)

Abstract - In this paper a new robotic system is presented. The main characteristic is represented by the modular and distributed hardware/software control architecture. The basic building block consists in a System-on-Module hardware component that is equipped with a processing unit (ST microcontroller), two Ethernet ports and several digital/analogical I/O ports. This basic component has been customized into two intelligent robotic modules, one for motor control and the other for intelligent vision. These modules are interfaced to the Player open source robotic middleware, in order to promote their interoperability with existing control application and to simplify the transition from a simulated environment (Gazebo) to the real system. They have been tested in a mobile manipulation scenario.

Keywords: Robotic middleware, modularity, reusability, standard interface, robotic simulator.

1 Introduction

In this paper we describe a new robotic system that has been built with particular attention on robotic hardware/software modularity and reusability. The experimental setup consists in a robotic platform (the AGV-Joker shown in Figure 1) built on COTS (Commercial Off The Shelf) components: a four independently actuated fixed-wheels rover and a six D.o.F. (Degrees Of Freedom) manipulator arm.

Workshop Proceedings of SIMPAR 2008
Intl. Conf. on SIMULATION, MODELING and PROGRAMMING for AUTONOMOUS ROBOTS
Venice(Italy) 2008 November,3-4
ISBN 978-88-95872-01-8
pp. 562-569

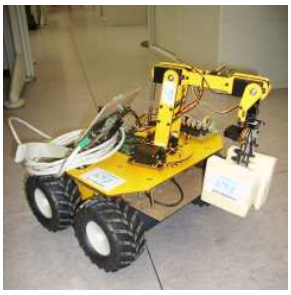

Fig. 1 The AGV-Joker robot used for testing

This platform is controlled by a network of intelligent hardware/software modules, which communicates using Ethernet links. Two intelligent modules have been currently developed: a motor control module and an intelligent vision system.

The networked control system can interconnect an arbitrary number of intelligent modules and master nodes.

The AGV-Joker robotic testbed is controlled by a network comprising a single motor control module, an intelligent vision system and a laptop running the main control program. Figure 2 shows a block diagram of the robot control network.

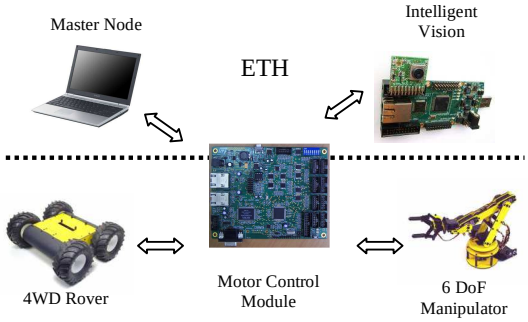

Fig. 2 Block diagram of the robotic system.

On the master node run a Player Server [1] and a client application written in C++ that talks with the intelligent modules through a Player's interface. The operating system is GNU/Linux and in particular a Kubuntu distribution. The communication interface is a Fast Ethernet Bus (100 Mbit/s).

The rest of the paper describes the hardware/software architecture of the intelligent modules and their interconnection with the Player middleware.

Workshop Proceedings of SIMPAR 2008
Intl. Conf. on SIMULATION, MODELING and PROGRAMMING for AUTONOMOUS ROBOTS
Venice(Italy) 2008 November,3-4
ISBN 978-88-95872-01-8
pp. 562-569

2 Intelligent Module Architecture

Robotic modules can be divided into two categories: smart modules and basic modules. Basic modules are devices without embedded intelligence that provide only raw data. Smart modules can provide raw data or run customizable algorithms and send high level information to the host processor.

An example of COTS smart module is the CMUcam [3]. This device is composed of an ARM7@60Mhz microcontroller and a CMOS camera sensor module.

In the field of motor control there are a multitude of modules with different features. The modules have different communication interfaces and often each producer implements a proprietary communication protocol. For this reason a user that is used to work with a particular device will not be able to switch to another one without deeply changing his software.

Two different robotic modules have been developed: one for controlling the motors and the manipulator arm and one for intelligent vision. These modules are easily reprogrammable by using an open source toolchain based on Eclipse IDE and MinGW compiler.

2.1 Motor control module

The motor control module (MC-ERT-188) is capable of controlling, in PWM, 8 DC motors with encoder feedback. Each axis can monitor the motor current and has two digital inputs, which can be used as limit switches in manipulator control applications. Each axis can be controlled in three different modalities: Position (PID, Trapezoidal Speed profile or Minimum Jerk); Speed (PID), Current and Open loop (the control loop runs outside the module). Moreover there are several free digital I/O, an eight channel DIP-switch and 8 secondary PWM (that can be used to control 8 RC servomotors). The module has also a MotCtrl port that can be used to control a brushless (BLDC) motor. The core of this module is a STMicroelectronics STR912FA microcontroller, with a powerful 32bit ARM9 CPU running at 96MHz, lots of RAM (up to 96KB) and Flash memory (up to 2MB) and several peripherals (including Fast Ethernet, CAN 2.0B and USB 2 FS).

Figure 3 shows a picture of the module.

Workshop Proceedings of SIMPAR 2008
Intl. Conf. on SIMULATION, MODELING and PROGRAMMING for AUTONOMOUS ROBOTS
Venice(Italy) 2008 November,3-4
ISBN 978-88-95872-01-8
pp. 562-569

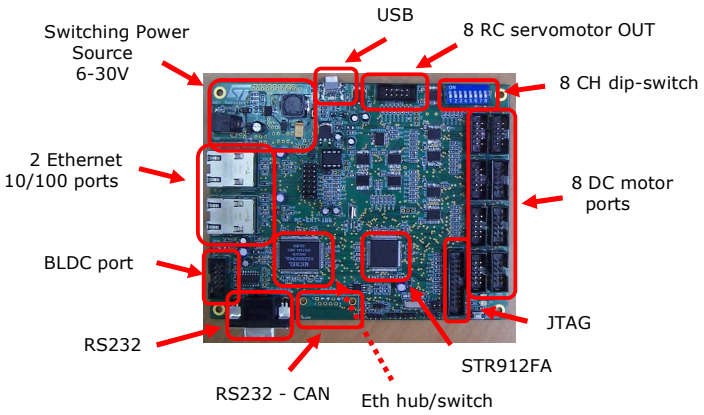

Fig. 3 The motor control module.

The microcontroller has got an eight channel DMA controller; two of them are used respectively to generate the 8 PWMs and to acquire the encoder signals. In order to reduce the overall CPU time required for the decoding of the encoder signals, an optimized assembly routine has been developed. Moreover the 32 bit CPU has been programmed to process in parallel the 8 encoder signals, i.e. like having eight four bit processors.
By using the DMA controller and by optimizing some routines in assembly language, the PWM generation at 24KHz - 8bit and the quadrature decode of the encoder signals acquired at 500KHz, required only 14% of CPU usage.

The module has two Fast Ethernet port (100Mb/s) managed by the Micrel KSZ8893-MQL hub/switch. The STR912FA microcontroller communicates with the physical layer by using a MII (Media Independent Interface) peripheral.

2.2 Vision system module

The second robotic module presented in this document is an embedded vision system based on the STMicroelectronics STR912FA [5] microcontroller (the same used for the motor control module) and the STMicroelectronics VS6724 camera (with a resolution of 2Mpx) [6].

The microcontroller is capable of processing a video stream in real-time or pass it to a host processor through its embedded peripherals (Ethernet, USB, CAN, RS232, I2C and SPI).

The microcontroller uses one DMA channel to acquire images, requiring less than 1% of CPU usage (this is due to an interrupt generated for each acquired row).

Table 1 shows the number of instructions per pixel available for user algorithms, computed at various resolutions and frame rates.

Workshop Proceedings of SIMPAR 2008
Intl. Conf. on SIMULATION, MODELING and PROGRAMMING for AUTONOMOUS ROBOTS
Venice(Italy) 2008 November,3-4
ISBN 978-88-95872-01-8
pp. 562-569

Table 1. Robotic vision system benchmarks

Image resolution	160x120	240x180	240x180
Frame rate	30	30	15
Acquisition overhead	0.41%	0.62%	0.31%
Image size (pixel)	19200	43200	43200
Instructions per image	3200000	3200000	6400000
Instructions per pixel	166	74	148

Common vision systems need to acquire completely an image prior to start the processing phase; in our system, instead, acquisition and processing can be done simultaneously row by row.

Thanks to the computational power of the microcontroller and to its embedded RAM and Flash, several vision algorithms have been tested on this platform. For example Figure 4 shows an off the shelf robotic platform (iRobot® Create™) that is driven by such intelligent vision node to identify and follow a red ball on the floor.

The possibility to send the video stream together with some processing results can

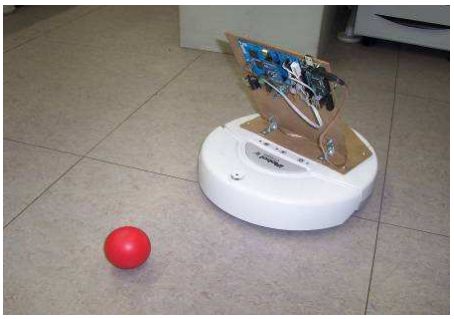

Fig. 4 Picture of the platform used to test the vision system module.

be used in a distributed computing architecture to reduce the computational power required to the host processor. For example, Figure 5 shows the results obtained by using an embedded image processing algorithm to find blobs of a predefined set of colours. The vision system module [4] is able to send the raw video stream together with the list and the parameters of the found blobs (centroid, perimeter, area, circularity and outer box).

Workshop Proceedings of SIMPAR 2008
Intl. Conf. on SIMULATION, MODELING and PROGRAMMING for AUTONOMOUS ROBOTS
Venice(Italy) 2008 November,3-4
ISBN 978-88-95872-01-8
pp. 562-569

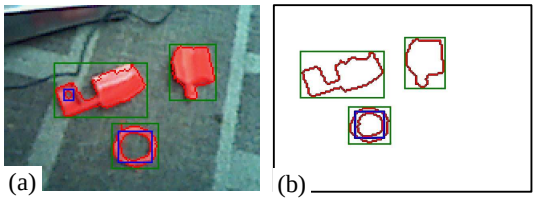

Fig. 5 Examples of real-time computation of the implemented smart camera with (a) and without (b)

Another important feature of this module is the presence of a synchronization port that can be used to synchronize several cameras (e.g. in order to do stereo vision).

Recently, a very small module (30mm x 30mm) has been developed by STMicroelectronics, this module includes an STR912FA microcontroller, a VS6724 camera and up to 16MB of external memory.

3 The Master Control Node

The high level software is based on a well known robotic middleware, i.e. Player-Gazebo. Player is a network server (over IP) which adds an abstraction layer to handle the robot with a standard interface. Gazebo is a 3D multi-robot simulator with dynamics. In order to interface the robot with Player, a custom built driver has been developed. In this way, it is possible to connect the GUI directly to the robot or to the 3D simulator in a transparent manner.

Our robotic modules can be easily controlled through a standard interface defined in the Player middleware. A custom client application can communicate with Player server through TCP socket; Player translates standard commands by passing them to the correct driver that, through the appropriate communication channel, forwards data to the real robot or to a simulator like Gazebo. In this way we add a layer of abstraction so that users can utilize their own client application by implementing few standard commands. For example in order to move a mobile robot the client application needs to send only speed or position commands.

Player server functionalities can be easily extended by using plug-in drivers. A driver is a piece of software (usually a class written in C++) that talks to a robotic sensor, actuator, or algorithm and translates its inputs and outputs to conform to one or more interfaces. An interface defines how to interact with a certain class. There are several standard interfaces already available in Player.

5 System reusability and reconfigurability

Player drivers allow the translation of standard high level command into hardware specific commands of the connected peripherals. In order to move a mobile robot, for

Workshop Proceedings of SIMPAR 2008
Intl. Conf. on SIMULATION, MODELING and PROGRAMMING for AUTONOMOUS ROBOTS
Venice(Italy) 2008 November,3-4
ISBN 978-88-95872-01-8
pp. 562-569

example, Player uses a *SetSpeed* command. This command has two arguments: linear and angular speed of the robot platform. Our driver must translate this command into speed commands for the four motors of the platform. All the parameters required for this translation are passed to the driver by using a special initialization file conforming to the specification of the Player project. In this way there is no need to recompile the driver in order to use it on a different platform where, for example, there are only two actuated wheels, or where transmission and kinematic parameters are different (i.e. encoder resolution, gear ratio, wheel diameter and wheel base distance).

The Intelligent Vision Module allows real time processing of video streams. Several vision algorithms have been already tested on this module. One for example allows the segmentation of images depending on color (several different RGB classes of color can be defined) and shape (perimeter, area and circularity) of found blobs.

If the robotic system requires a more complex vision algorithm, the module can process some filtering algorithms and stream the resulting images to a host PC performing the rest of the elaboration.

The network employed [2] is based on the Fast Ethernet standard, so to exploit all the advantages of such a widespread interface, including switches and wireless routers.

4 Conclusions

In this paper a new robotic system has been presented, the robot is controlled by using two main robotic modules (one for motor control and the other for intelligent vision), specifically designed to be modular, scalable and reusable for other applications.

These modules are interfaced to a well known and open source robotic middleware, Player, by using two drivers. Player offers an abstraction layer to the hardware and it is compatible with an opens source multi-robot 3D simulator, Gazebo. This allows the design of a GUI for a generic robot, which can be tested on the simulator, on our robotic system or on any other robotic system running Player.

References

1. Toby H.J. Collett, Bruce A. MacDonald, and Brian P. Gerkey, "Player 2.0: Toward a Practical Robot Programming Framework". In Proceedings of the Australasian Conference on Robotics and Automation (ACRA 2005), Sydney, Australia, December 2005
2. D. Caltabiano, D. Brugali, R. Sannino, D. Ghezzi, L. Spelgatti, “A Real-Time communication protocol for interconnecting robotic smart devices”, accepted for presentation and publication in the Proceedings of the 2008 IEEE International Conference on Intelligent Robots and Systems (IROS 2008) to be held in Nice, France, on September 22-26, 2008.

Workshop Proceedings of SIMPAR 2008
Intl. Conf. on SIMULATION, MODELING and PROGRAMMING for AUTONOMOUS ROBOTS
Venice(Italy) 2008 November,3-4
ISBN 978-88-95872-01-8
pp. 562-569

3. A. Rowe, A.G. Goode, D. Goel, and I. Nourbakhsh, “CMUcam3: An Open Programmable Embedded Vision Sensor”, tech. report CMU-RI-TR-07-13, Robotics Institute, Carnegie Mellon University, May, 2007.
4. V. Rana, M. Matteucci, D. Caltabiano, R. Sannino and A. Bonarini, “Low cost smartcams design”, accepted for publication in the Proceedings of the 6th IEEE Workshop on Embedded Systems for Real-Time Multimedia (ESTIMedia 08), to be held in Atlanta, Georgia, October 23-24, 2008.
5. http://www.st.com/mcu
6. http://www.st.com/imaging

Workshop Proceedings of SIMPAR 2008
Intl. Conf. on SIMULATION, MODELING and PROGRAMMING for AUTONOMOUS ROBOTS
Venice(Italy) 2008 November,3-4
ISBN 978-88-95872-01-8
pp. 562-569