Design and implementation of a multi-sensor module for mobile robotics applications

Gonzalo F. Perez Paina, Fernando E. Elizondo, David A. Suarez, Luis R. Canali
IT Research Centre (CIII)
National Technological University, Córdoba Regional Faculty (UTN-FRC)
Email: gperez@scdt.frc.utn.edu.ar

***Abstract*—This paper presents the design and implementation of a sensor module for mobile robotics applications. Even though the module was designed for a specific mobile robot it can be adapted to different types of robots, either wheeled or flying vehicles. The set of sensors comprises an ultrasonic ring allowing the robot to perceive its surrounding environment, and an inertial sensor useful for motion estimation in data integration approaches. The present paper describes in detail the design of an embedded system for the sensor module including the hardware, firmware and high level processing software framework.**

I. Introduction

One of the main tasks to be performed by autonomous systems is to acquire knowledge about the operational environment. This is done using different types of sensors. For mobile robotics applications there is a wide variety of sensors, to obtain either internal measurements or information about the surrounding environment. These sensors can be classified as proprioceptives/exteroceptives and actives/passives [1].

Proprioceptive sensors are used to measure internal robot parameters, such as the speed of drive motors, wheel's load, battery voltage, etc. Exteroceptive sensors are used to acquire external information about the surrounding environment, such as distance measurements, light intensity, etc.

In order to get useful environmental information, the robot must *interpret* exteroceptive sensor measurements. On one hand, passive sensors measure energy coming from the environment, e.g. temperature sensors, microphones, CCD or CMOS cameras, etc. On the other, active sensors emit energy into the environment and measure how this energy returns to the sensor altered by the environment, e.g. ultrasonic sensors, laser range finders, infrared, etc. Given that this type of sensors allows controlling the interaction with the environment, they generally offer better performance than passive ones despite the fact that these signals can suffer a wide variety of interferences.

One of the most used sensor for mobile robots, which is generally part of commercial robots, is an ultrasonic ring. The measurements of ultrasonic sensors are used by robotics algorithms in several tasks like obstacle avoidance [2], localization [3][4], mapping [5], SLAM [6], etc. Other common sensors in robotics are inertial sensors which are useful in many navigation algorithms, for instance to improve the localization of a mobile robot by means of fusing the wheel odometry data with inertial measurements [7][8]. Moreover, it is also necessary to obtain the robot attitude in rough terrain applications [9] or flying vehicles, applying filtering techniques that integrate all measurements like those presented in [10].

Fig. 1. RoMAA robot with ultrasonic ring.

The Open Architecture Mobile Robot (RoMAA in Spanish) [11] is a robot designed and built entirely in the Information Technology Research Centre (C.I.I.I. in Spanish) as an experimental vehicle intended for use in research fields of autonomous mobile robotics and computer vision. The high level control processing of the robot is carried out by an onboard PC which uses the robotic development environment Player [12]. This robotic framework allows to interact with robotic sensors and actuators in a client/server architecture over TCP/IP networks. Furthermore, it provides a hardware abstraction layer of robotic devices. The present paper describes the design and implementation of a complete embedded system of a sensor module for the RoMAA robot. This module is composed of an ultrasonic ring along with an inertial sensor based on MEMS (Microelectromechanical System) devices, and can be adapted to different types of robotic architectures, either wheeled or flying vehicles [13]. In Fig. 1 the RoMAA robot with the ultrasonic ring is shown.

This paper is organized as follows. Section II gives an overview of the characteristics and features of the developed sensor module. Section III details the design and implementation of the module, and section IV presents the main features of the module as a complete system capable of being adapted to other robots. Conclusions and future works are presented

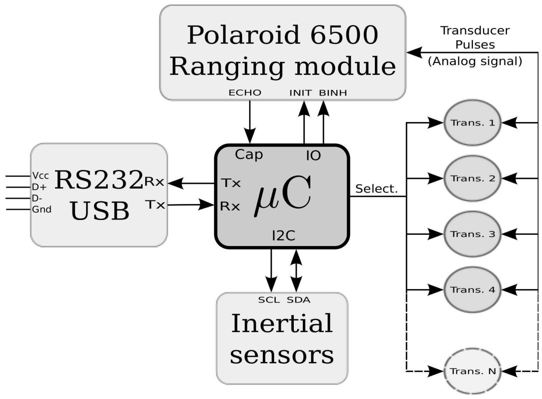


Fig. 2. Block diagram of the developed sensor module.

in section V.

II. General description

The sensor module presented in this paper is composed of two parts: an ultrasonic ring and an inertial measurement unit. Fig. 2 shows the system block diagram. The embedded system is based on a microcontroller of the ARM7-TDMI family, more specifically one of the LPC21XX series by NXP [14]. The microcontroller is the central part of the sensor module. The development of the firmware was based on embedded libraries specifically designed for robotic applications [15].

The ultrasonic ring is composed of nine transducers which are multiplexed in order to operate all together with only one ranging module. The microcontroller manages all the signals of the ranging module as well as the multiplexor. The inertial sensor part is based on an OEM module (9 Degrees of Freedom-Sensor Stick) composed of different MEMS devices, which communicates with the microcontroller using an I^2C bus (Inter-Integrated Circuit).

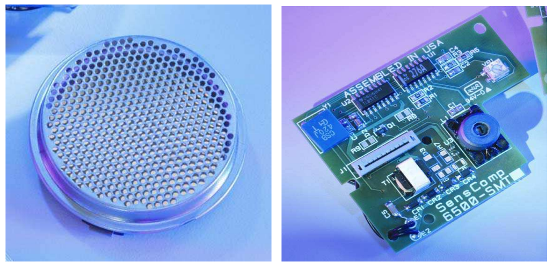
The sensor module communicates with the robot's on-board processing computer through an USB port. The PC reads the sensorial information using the Player [1] robotic development environment. In order to be able to use this framework the appropriate Player driver was also developed providing the abstraction from the low level hardware implementation.

III. Sensor module

A. Ultrasonic ring sensor

The developed ultrasonic ring is composed of nine electrostatic transducers of SensComp Inc. More specifically the used transducers are of the 600 series environmental transducer, Fig. 3(a). These operate as both transmitter and receiver of the ultrasonic wave. The transducer emits the ultrasonic wave to the environment and the reflected waves from nearby surfaces are also measured by the same device. The emitted ultrasonic wave lobe is similar to a conic shape of approximately $\pm 15°$@-6dB. The transducer can measure distances from 0.15m to 10.7m, with a resolution of $\pm 1\%$ of full range (e.g. ± 3mm at 3m).

[1] http://playerstage.sourceforge.net/



(a) Transducer 600 series (b) 6500 Ranging Module

Fig. 3. Components of the ultrasonic ring of SensComp Inc.

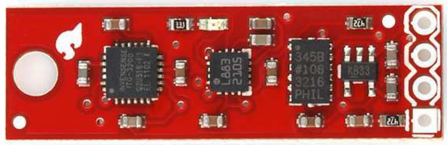


Fig. 4. 9 Degrees of Freedon-Sensor Stick IMU of Sparkfun.

The transducers are driven by the "6500 Enhanced SMT Ranging Module", also of SensComp Inc., Fig. 3(b). This module, working in a multiple echo mode, allows to detect near objects at distances of only 7.3cm.

The ranging module has TTL input and output signals which communicates with the microcontroller, and an analog signal which interacts with the transducer. The former are: INIT (Initialize Input), BINH (Blanking Inhibit), BLNK (Blanking), all input signals, and the ECHO (Echo return output) signal. The ranging module has two operational modes, the normal and the multi-echo mode whose main difference is in the use of BNLK signal. In the present work, the ranging module has been used in normal mode, in which this signal has to be permanently in low level. In addition, the ranging module generates an "internal blanking" signal of 2.38ms, which allows to measure a minimal distance of 0.4m. Given that the used transducers are able to measure distances from 0.15m, the implemented hardware uses the BINH signal, disabling the "internal blanking".

Fig. 5 shows a time diagram of the ranging module signals. The microcontroller starts the measurement which is indicated by the rising edge of the INIT signal. This makes the ranging module generate a burst in the analog signal to the transducer producing the ultrasonic wave (this is considered the initial time t_0). After a time of 0.87ms (corresponding to a measured distance of 0.15m) the microcontroller generates a rising edge in the BINH (without waiting this minimum time, the ECHO signal of the ranging module is not valid). The ultrasonic wave is reflected by a surface in the environment which makes the wave to return. This makes the ranging module to generate the ECHO signal which enables the microcontroller to obtain the measurement. The distances d is computed using

$$d = \frac{v_s t}{2} \quad (1)$$

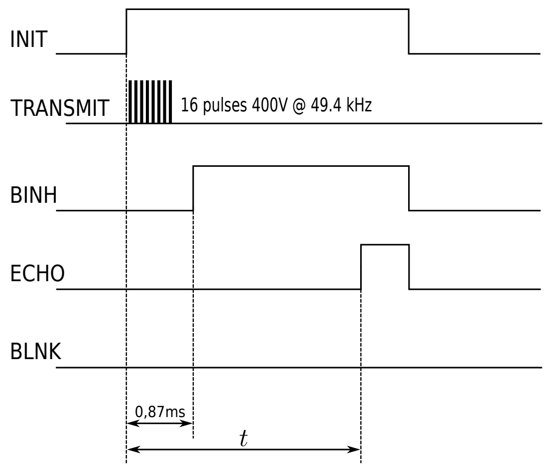


Fig. 5. Time diagram of the ranging module signals.

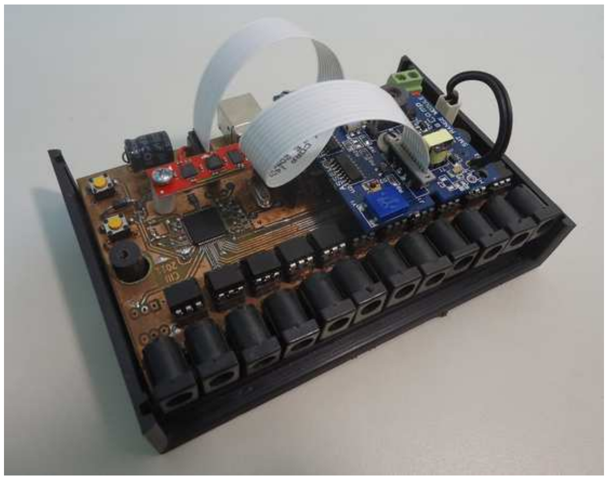


Fig. 7. Developed sensor module for ultrasonic ring and inertial sensor.

where $v_s(= 342.2\text{m/s}@20°C)$ is the speed of sound and t the time measured by the microcontroller from t_0 till the echo signal.

In order to operate with more than one transducer, all signals of the ranging module are multiplexed. The multiplexor works by choosing the transducer to be used, after which the microcontroller shoots the ultrasonic wave using the INIT signal of the ranging module, and obtains the measurement through the ECHO signal. Both signals, the output (16 pulses of 400V@49.4KHz) and the input (ECHO signal) form a common bus to all the transducers. Each transducer is selected using photovoltaic relays model PVU414, which are commanded by the microcontroller. Fig. 6 shows in detail the designed schematic circuit used in this module part.

The circuit was designed to operate with a total of 12 transducers. This allows to add extra transducers which could be used to measure the environment backward the robot.

B. Inertial sensor

For the inertial measurement unit it was chosen to used the OEM 9 Degrees of Freedom - Sensor Stick (sku: SEN-10724) of Sparkfun®, Fig. 4. This module is released under Creative Commons licence, and is built based on MEMS devices, including accelerometers, gyroscopes, and magnetic compasses. The module has an I^2C communication bus. The more important sensor characteristics are:

- Analog Devices accelerometer model ADXL345.
  - 3-axis accelerometer.
  - 4 selectable ranges of ±2g, ±4g, ±8g and ± 16g.
  - High resolution of 4mg/LSB in all ranges.
  - 10 bits of minimum resolution, and up to 13 bits at full scale (±16g).
- InvenSense Gyroscope model ITG3200.
  - 3-axis gyroscope.
  - Full scale range of ±2000°/sec.
  - Sensitivity of 14.375 LSBs per °/sec.
  - 3 different ADC (Analog to Digital Converter) of 16 bits.
- Magnetic compass Honeywell model HMC5883L.
  - 3-axis magnetoresistive sensor.
  - Full scale range of ±8 gauss.
  - Resolution of 2 milli-guass.
  - 12 bits ADC, enables 1° to 2° compass heading accuracy.
  - Designed to measure both the direction and the magnitude of Earth's magnetic fields.

Before being able to use the inertial measurements it is necessary to carry out a calibration process to obtain correction parameters. These parameters correspond to inherent sensor errors like non-zero bias, non-unit scale factor [16][17], non-orthogonal misalignment of the sensor axes [18] and the cross-axis sensitivity. Additionally, it is also necessary to know the rigid transformation between the robot and the sensor unit coordinate systems. The calibration process of the inertial measurement unit is not presented here and is left as future work.

IV. Module operation

A. Low level - Embedded system

The developed sensor is shown in Fig. 7. The module communicates with the high level processing on-board PC using an USB connection. The communication frame is shown in Fig. 8. The `DATA` part of the frame corresponds to the useful data package, which consists of one command byte followed by a non mandatory data part depending on the specific command. Commands are used to write and read configuration parameters, or to read sensory data.

In Tab. I the implemented commands are listed. Some of the most relevant commands for the ultrasonic ring are:

- `CMD_SET/GET_RADIUS`, are used to write and read the distances r_i (`RADIUS`) from the ultrasonic ring centre to all transducers.

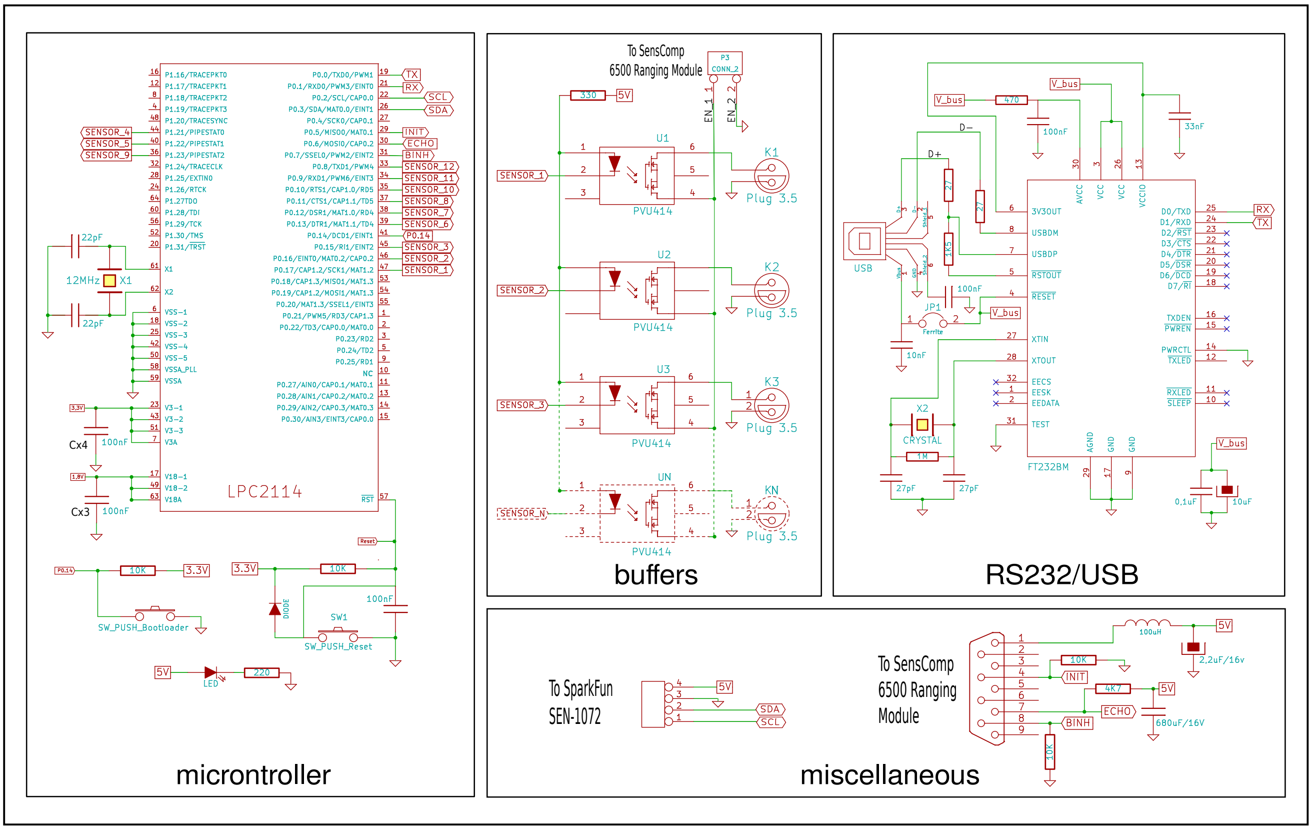


Fig. 6. Schematic circuit of the developed sensor module.

- CMD_SET/GET_ANGLES, are used to write and read the angles φ_i (ANGLE) from the axis of the front sonar to all transducers.
- CMD_SET/GET_SON_POS, are used to write and read the position (x, y, z) of the sonar coordinate system w.r.t. the robot coordinate system.

Fig. 9 shows the geometric distribution of all sonar transducers. Setting the geometry of the ultrasonic ring can be used to set the position (r_i, φ_i) of all transducers measured after being attached to the mechanical structure. Moreover, the position (x, y, z) of the sonar coordinate system with respect to the robot coordinate system can be "fine tuned" after a calibration procedure. This procedure is done by moving the robot through known trajectories and comparing the distance measurements of the sonar to fixed obstacles.

Some of the most relevant commands for the inertial sensors are:

- CMD_SET/GET_IMU_CALIB, are used to write and read calibration parameters (this command is reserved for future works related to IMU calibration).
- CMD_SET/GET_IMU_POSE, are used to write and read the position and orientation (six parameters) of the IMU coordinate system w.r.t. the robot coordinate system.
- CMD_GET_IMU_ANGLES, is used to read the estimated orientation of the IMU by fusing the inertial sensor data (this command is received for future works related to inertial sensor fusion).

The module also implements logging commands which are

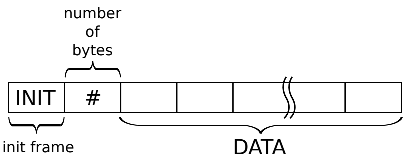


Fig. 8. Communication frame of the sensor module.

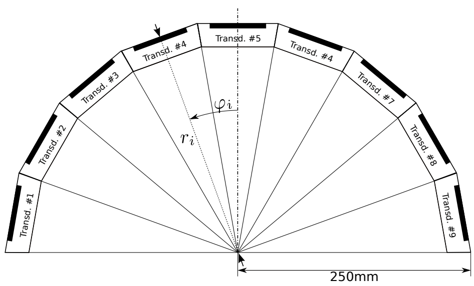


Fig. 9. Geometric distribution of sonar transducers (not in scale).

Configuration and geometric commands	
CMD_SET_COUNT	Set number of sonar transducer
CMD_GET_COUNT	Get number of sonar transducer
CMD_GET_MIN_RANGE	Get sonar minimum range
CMD_GET_MAX_RANGE	Get sonar maximum range
CMD_SET_RADIUS	Set radius of sonar transd. w.r.t. SCS
CMD_GET_RADIUS	Get radius of sonar transd. w.r.t. SCS
CMD_SET_ANGLES	Set angles of sonar transd. w.r.t. SCS
CMD_GET_ANGLES	Get angles of sonar transd. w.r.t. SCS
CMD_GET_SON_POS	Get (x, y, z) sonar ring position w.r.t. RCS
CMD_SET_SON_POS	Set (x, y, z) sonar ring position w.r.t. RCS
CMD_SET_IMU_CALIB	Set IMU calibration parameters
CMD_GET_IMU_CALIB	Get IMU calibration parameters
CMD_SET_IMU_POSE	Set IMU relative pose w.r.t. RCS
CMD_GET_IMU_POSE	Get IMU relative pose w.r.t. RCS
Data commands	
CMD_GET_RANGE	Get range of one specific transducer
CMD_GET_SCAN	Get ranges of all sonar transducers
CMD_GET_IMU_ANGLES	Get IMU orientation data
CMD_GET_IMU_RAW	Get all IMU sensors raw data
CMD_GET_ACC_RAW	Get accelerometer raw data
CMD_GET_GIR_RAW	Get gyroscope raw data
CMD_GET_COM_RAW	Get magnetic compass raw data
Logging commands	
CMD_SET_LOG_TIME	Set log time in milliseconds
CMD_SON_LOG_INIT	Init sonar data logging
CMD_IMU_LOG_INIT	Init IMU data logging
CMD_LOG_INIT	Init sonar and IMU data logging
CMD_LOG_STOP	Stop data logging

TABLE I
IMPLEMENTED COMMANDS OF THE SENSOR MODULE. (RCS: ROBOT COORDINATE SYSTEM - SCS: SONAR COORDINATE SYSTEM).

very useful in robotic applications. In data logging mode the microcontroller sends periodically sensor data to the on-board PC. These logging commands include setting the log rate (specified in milliseconds), and starting/stopping data logging either of the ultrasonic ring, the inertial measurement unit or both together.

B. High level - PC software

The on-board PC software has been developed to be used in the GNU/Linux operating system. This includes a communication library to send and receive data to and from the sensor module, and a specific driver for the Player robotic development environment.

The communication library allows the transmission and reception of sensor module data without having to know the low level communication protocol, and has been developed using the `flexiport` library of the Gearbox project [2]. The communication library is based on a C++ class (in the OOP sense) and implements similar methods for each of the communication commands of the embedded system. For instance, the methods `get_min_range`, `get_max_range`, `get_scan`, `get_range` for the ultrasonic ring, and `set_imu_pose`, `get_imu_pose`, `get_imu_raw` for the inertial sensor unit.

The high level control software for the robotic platform that processes all the sensory data is developed using the Player robotic development environment [12]. Player works as a hardware abstraction layer (HAL) allowing the access to robotic devices (sensor and actuators) in a simple and transparent way. It is based on sockets working in TCP/IP networks. The socket abstraction enables independence of both the programming language and the computer platform. Player is built in two parts, one is the network server running in the robot on-board PC, and the other is composed of remote client applications. These client applications are built using client libraries included in the Player development environment. Available client libraries are implemented in C, C++ and python programming languages. Client applications communicate with the server to access to the robotic devices using predefined interfaces. These interfaces specify how to interact with each class of robotic devices. A schematic representation of the different layers of abstraction using Player can be seen in Fig. 10.

[2] `http://gearbox.sourceforge.net/`

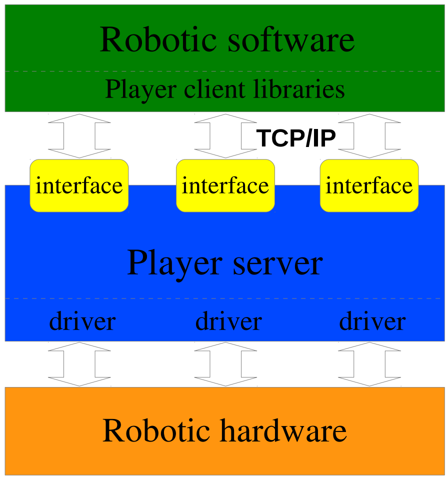


Fig. 10. Player working scheme and relation with the hardware.

In order to use the presented sensor module with the Player robotic development environment it was needed to write the appropriate Player driver. This driver bounds the low level device implementation with Player predefined interfaces. The used interfaces are `ranger` for the ultrasonic ring and `imu` for the inertial sensors. The Player driver is implemented using libraries included in the development environment, and is coded as a C++ class. A detailed description of the implementation of a custom driver is shown in [19].

The remote client applications have access to the sensor module data by means of proxy objects, which are `RangerProxy` and `ImuProxy` for the ultrasonic ring and the inertial measurement unit, respectively. These proxies have methods like `GetRangeCount`, `GetRange`, `RequestGeom`, `GetMinRange`, `GetMaxRange`, `GetRangeRes` for the `RangeProxy`; and `GetPose`, `GetX/Y/ZAccel`, `GetX/Y/ZGyro`, `GetX/Y/ZMagn`, `GetRawValues` for the `ImuProxy`.

V. Conclusions and future works

The design and implementation of a sensor module for mobile robotics applications was presented. It is composed of an ultrasonic ring and an inertial measurement unit. The complete embedded system was described in detail, from the low level hardware to the high level software. The described items were the designed hardware (with details on used components), firmware, low level communication, and the high level communication library (a C++ class) written to access the sensor data. Moreover, in order to use the adopted Player robotic development environment, it was also needed to write a specific driver which bounds the hardware implementation with generic Player interfaces. This allows the use of the sensor module with this powerful robotic framework.

One of the main goals taken into account in the design was to obtain a flexible and easily configured sensor module, that was able to be adapted to different experiments in the robotics research field, even to other robot architecture. Given that the embedded system can be configured using the implemented command, and the modularity of the developed software running in the on-board PC, it is considered to have achieved the proposed goals. It is important to note that the modularity of the robot's on-board software allows to build robotic applications using the communication library directly or the Player robotic development environment together with the developed Player driver.

The developed sensor module has characteristics that are similar to the typical sensors included in commercial mobile robots. This module is now being used in robotic applications for research purposes, mainly the ultrasonic ring as a position reference.

Future works include research and experimentation on IMU calibration methods and sensor fusion techniques in order to improve the odometry obtained from wheel encoders.

Acknowledgements

The first author of the paper is financed by a grant from the program of PhD scholarships of the National Ministry of Science and Technology of Argentina through its financing Agency. The work presented in the paper is financed by the University project U.T.N. PID 1151 "Open Architecture Mobile Robot, RoMAA-II", SCyT-UTN Nº34/10.

References

[1] Roland Siegwart and Illah R. Nourbakhsh. *Introduction to Autonomous Mobile Robots*. Bradford Company, Scituate, MA, USA, 2004.

[2] A. Ferreira, F.L. Pereira, T.F. Bastos-Filho, M. Sarcinelli-Filho, and R. Carelli. Avoiding obstacles in mobile robot navigation: Implementing the tangential escape approach. In *Industrial Electronics, 2006 IEEE International Symposium on*, volume 4, pages 2732 –2737, july 2006.

[3] Jinwoo Choi, Minyong Choi, Il Hong Suh, and Wan Kyun Chung. Topological localization using sonar gridmap matching in home environment. In *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, pages 4910 –4915, may 2010.

[4] Antoni Burguera, Yolanda González, and Gabriel Oliver. Sonar sensor models and their application to mobile robot localization. *Sensors*, 9(12):10217–10243, 2009.

[5] Se Jin Lee, Dong Woo Cho, Wan Kyun Chung, Jong Hwan Lim, and Chul Ung Kang. Feature based map building using sparse sonar data. In *Intelligent Robots and Systems, 2005. (IROS 2005). 2005 IEEE/RSJ International Conference on*, pages 1648 – 1652, aug. 2005.

[6] Juan D. Tardós, José Neira, Paul M. Newman, and John J. Leonard. Robust mapping and localization in indoor environments using sonar data. *Int. J. Robotics Research*, 21:311–330, 2002.

[7] J. Borenstein and L. Feng. Gyrodometry: A new method for combining data from gyros and odom etry in mobile robots. In *In Proceedings of the 1996 IEEE International Conference onRobotics and Automation*, pages 423–428, 1996.

[8] N. Houshangi and F. Azizi. Accurate mobile robot position determination using unscented kalman filter. In *Electrical and Computer Engineering, 2005. Canadian Conference on*, pages 846 –851, may 2005.

[9] J. Vaganay, M.J. Aldon, and A. Fournier. Mobile robot attitude estimation by fusion of inertial data. In *Robotics and Automation, 1993. Proceedings., 1993 IEEE International Conference on*, pages 277 –282 vol.1, may 1993.

[10] G. Perez Paina, D. Gaydou, J. Redolfi, C. Paz, and L. Canali. Experimental comparison of kalman and complementary filter for attitude estimation. In *Jornadas Argentinas de Informática*, pages 205–215. AST, Argentine Symposium on Technology, 2011.

[11] D. A. Gaydou, G. F. Pérez Paina, G. M. Steiner, and J. Salomone. Plataforma móvil de arquitectura abierta. In *Proceedings of the V Argentine Symposium of Robotics 2008*. Ediuns, 2008, November 2008.

[12] Toby H. J. Collett, Bruce A. Macdonald, and Brian P. Gerkey. Player 2.0: Toward a practical robot programming framework. In *Proc. of the Australasian Conf. on Robotics and Automation (ACRA)*, Sydney, Australia, 2005.

[13] David Gaydou, Javier Redolfi, and Agustín Henze. Filtro complementario para estimación de actitud aplicado al controlador embebido de un cuatrirrotor. In *Congreso Argentino de Sistemas Embebidos*, 2011.

[14] NXP Semiconductors. *UM10114. LPC21xx and 22xx User Manual*, April 2007.

[15] Gonzalo F. Perez Paina, David A. Gaydou, Néstor L. Palomeque, and Lucas A. Martini. Librerías embebidas para microcontroladores lpc2000 de aplicación en robótica. In *Congreso Argentino de Sistemas Embebidos*, 2011.

[16] Eduardo Nebot and Hugh Durrant-Whyte. Initial calibration and alignment of low-cost inertial navigation units for land vehicle applications. *Journal of Robotic Systems*, 16(2):81–92, 1999.

[17] lukowicz, Paul. Automatic Calibration of Body Worn Acceleration Sensors. In a, editor, *PERVASIVE 2004, LNCS 3001*, 2004.

[18] Albert Krohn, Michael Beigl, Christian Decker, Uwe Kochendörfer, Philip Robinson, and Tobias Zimmer. Inexpensive and automatic calibration for acceleration sensors. *Ubiquitous Computing Systems*, pages 245–258, 2005.

[19] Gonzalo F. Perez Paina and David A. Gaydou. Programación y simulación en robótica móvil utilizando player/stage. In *VI Jornadas Argentinas de Robótica*, 2010.