EURASIP Journal on Embedded Systems

RESEARCH Open Access

Embedded solutions for a class of highly unstable, underactuated and self-balancing robotic systems

Andrea Bonci*, Massimiliano Pirani and Sauro Longhi

Abstract

This paper presents a didactic framework in embedded electronics systems that is used to elicit awareness into students and engineers on the design issues arising in the realization of a class of underactuated robots and aerial vehicles that needs be robustly controlled due to their intrinsic unstability. The applications prototyped on the embedded platform presented here are conceived, by design, to be compliant with tiny collaborative robotics applications in order to adhere to the needs of the complex cyber-physical systems problem. The proposed platform is self-contained with on-board sensing and computation. Its engineering uses only off-the-shelf and mass production components. The system is based on a general purpose embedded board equipped with a 32-bit microcontroller which is able to manage all the basic tasks of this robotic platform: sensing, actuation, control and communication. The framework is described, and initial experimental results are introduced. Three applications are presented in this work as a validation of the methodology: a ballbot robot, a legged robot and a quadrotor aerial vehicle. The chosen case studies are robotics applications that are specialized in performing maneuvers when operating in tight spaces as in the human living environments.

Keywords: Microcontrollers, Collaborative robotics, Cyber-physical systems

1 Introduction

During the last decade, strong interests have been observed in the development of self-balancing and unmanned robots and vehicles. Self-balancing robots, unmanned aerial vehicles (UAV) , antropomorphic systems (legs, arms) can be considered within the interested class of robots and vehicles which are underactuated and intrinsically unstable. The interest in these kinds of applications has been stimulated by recent advances on embedded systems technology. Nowadays, embedded electronics systems represent the greatest part of existing computing devices. These devices have grown in performance and have lowered in costs and power consumption while featuring greater reliability and dependability. They currently represent the most important part in any control system technology application. Demanding and complex algorithms can be deployed on the embedded devices that implement them effectively through innovative hardware, advanced on-chip peripherals and software architecture provisions. For example, in [1], authors faced the problem of rapid prototyping of complex differential equation solvers that are used in most model-based control methods and their deployment on embedded systems electronics for mechatronics applications.

An embedded system is an application-specific computer that interacts with the physical world and often communicates with heterogeneous computing systems. Indeed, embedded electronics is a key component in cyber-physical system (CPS) applications that participate in a new momentum of growth of cybernetics technology. Cyber-physical systems encompass the technologies that confront with a new model of complexity in highly networked applications. Their design issues cannot be neglected in the engineering of novel robotics systems when a model-based design is advisable [2]. Currently, new computation environments are realized through embedded electronics, the hardware that sports

*Correspondence: a.bonci@univpm.it
Universita' Politecnica delle Marche, Dipartimento di Ingegneria dell'Informazione (DII), Via Brecce Bianche, 60131 Ancona, Italy

Springer Open

© 2016 Bonci et al. **Open Access** This article is distributed under the terms of the Creative Commons Attribution 4.0 International License (http://creativecommons.org/licenses/by/4.0/), which permits unrestricted use, distribution, and reproduction in any medium, provided you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license, and indicate if changes were made.

numerous miniaturized and integrated mechanical and electronics parts. Such a hardware infrastructure allows to communicate with all the sensors and actuators that are usually installed in a mobile robot. Embedded systems are capable of communication, actuation, control, sensing, signal processing, (inter)networking and human user interfaces.

The focus of this paper will be on the embedded systems technology to be adopted in controlled underactuated and intrinsically unstable systems having a limited number of actuators.

In order to self balance and move autonomously, the robots and the aerial vehicles need a well-engineered computational framework for an effective and cheap implementation on embedded electronics systems. The major part of the self-balancing and aerial vehicles are underactuated systems. This means that they have less actuated inputs than degrees of freedom. This is a clear advantage for the realization costs but generates complexity in computing, sensing and control.

In this paper, a unified hardware and software framework is proposed for a class of underactuated robots an vehicles. The architecture here presented leverages, low cost embedded systems for the control of up to four motors and with very limited use of encoders, through an effective exploitation of the advanced peripheral capabilities and hardware architectures featured by state-of-the-art systems on chip.

Mobile robots have a broad set of applications and have been playing an increasingly important role over the last few years. Nowadays, there are also several kinds of mobile robots capable of agility and complex movements. Some of them are anthropomorphic systems with their high level of complexity due to the high number of actuators. With increasing the actuators, numbers of course, we find costs, energy consumption and dependability issues arising. Nonetheless, the study of systems with a fewer number of actuators has been in progress. Among them, the most commercialized and known is the Segway®. This kind of system consists of a self-balanced two-wheeled robot. This vehicle allows a higher durability with respect to the anthropomorphic systems. Another system that balances and moves on a spherical wheel and that needs just the space of its body span for the movements is the ballbot system. It is an alternative locomotion method to two-wheeled mobile robots or to statically stable wheeled mobile robots.

In Section 2, we present some related works and literature about the three applications here tested and give some discussion hints about the different approaches used here with respect to the state-of-the-art. In Section 3, we present the case studies and their peculiarities form the control point of view. In Section 4, the overall methodology used is presented. It deals mainly with the exposition of useful and practical guidelines. The choice criteria for suitable embedded hardware is discussed along with the design constraints in a general purpose platform able to be reused and configured across the three chosen applications and then destined to be extended to a wider class of similar applications. In Section 5, we provide some measurements and results obtained from the prototypical implementations of the three robots. Section 6 is devoted to the exposition of the conclusion and the discussion of further work on the theme.

2 Related works

The three case studies provided here have been already studied elsewhere, though with a different extent and with different purposes. The first ballbot was developed in 2006 at Carnegie Mellon University (CMU) in the USA ([3, 4]). Another one has been developed by Tohoku Gakuin University (TGU) in Japan [5] and a third one was developed as student project in the University of Adelaide (UA) in Australia [6]. After then, several works have been developed about this topic. The proposed design work can be considered as an intermediate level between the other existing ballbot implementations. Indeed, the present work does not rely on high-level performance like in [7] and it is not implemented to explore new underactuated systems like in [3–5]. Our aim is to reach educational and research purposes mostly like in [6]. On purpose, we selected the components to make the student or the engineer able to explore the major issues in embedded cyber-physical systems, not only in the modeling and control part, but also in the software design part under real-time constraints.

Readers interested in historical review of legged robotics are referred to [8–10]. A complete overview on different actuation principles and their application in legged systems can be found in [11].

Once the mechanical platform for the robotic leg is established, a suitable embedded system must be chosen based on its requirements. The present paper focuses on the class of robotic legs actuated by electric DC motors. In particular, its goal is on the development of an embedded system designed and devoted to the low-level control of a robotic leg. The DC motors are particularly suitable for actuating legged robots thanks to their simplicity and dependability. A reasonable number of works can be found in the literature regarding the control of DC motors. Obviously, the idea of DC motor control is not new. The novelty and the focus of this paper is on the embedded architecture which is tailored to control all the motors of the leg's joints with a single low-level and low-cost controller constituted by a single embedded electronics board. Furthermore, the choice of actuating the DC motors by means of DC drivers, makes the controller able to act on motors with some freedom in the range of the power specifications. Usually, in similar works, as [12], expensive position controllers have been used for each motor. This means that a single leg requires three controllers rather than one, and a total of twelve are involved for a quadrupedal robot instead of four. The computational power available on the proposed embedded board makes it viable to be used for high-level controllers for the management of inertial sensors, for generating the leg's trajectories and for coordinating movements without the exclusive use of high-cost devices. Depending on the desired leg's task (jumping, walking, running, etc.), actuators may require different control methods as position, velocity or torque control. Due to its hierarchical architecture, based on the motor model, the proposed low-level controller ensures high flexibility in switching between the available control methods for the leg's joints. Finally, it ensures also compactness, lower costs and less power consumption.

Concerning the quadrotor and aerial vehicles, they usually represent an example of highly unstable and nonlinear system. Unmanned aerial vehicles (UAVs) are among the most appealing topics in control applications on embedded systems. This research area is multidisciplinary and involves disciplines like electronics, mechanics, aeronautics, automatic control, signal processing, etc. Many groups have worked on standard quadrotors and realized various controller scenarios based on these platforms. Usually, UAV design implementations have been developed for proving the feasibility of concept. In many implementations, the embedded software is written specifically for a particular mix of sensors and actuators. A survey of embedded control systems for UAVs can be found in [13, 14].

3 Problem description

The aim of this paper is to propose a hardware and software framework that is self-contained with on-board sensing and computation and that can be applied to vehicles and robots of different nature and having in common the characteristics of being underactuated, intrinsically unstable and with a limited number of actuators. Such a framework relies mostly on easy retrievable off-the-shelf components. The number of actuators is limited by the microcontroller board performances. In the considered case, it is limited to four brushless motors without encoders or three DC motors with encoders.

On purpose, we selected the components to make the student or engineers proficient in exploring the major issues in embedded cyber-phisical systems, not only on the model and the control part but also on the software design part, under real-time constraints. For our educational goal, we took the YRDKRX63N board by Renesas, which can be used to challenge designers with cross-compiling issues, device driver design, communication protocols and real-time processing.

The three case studies considered are a ballbot system, a robotic leg prototype and a quadrotor aerial vehicle.

3.1 Didactic perspective of the work

From an educational point of view, the three case studies were suitable for multidisciplinary working teams. For each case, laboratory activities have been provided and divided into subtasks associated to the design phases of the robotic systems. The first task is the mechanical layout of the robotic systems. It should be defined by the team under common design constraints like the type of materials, the limit weight, the distribution of weights in order to obtain the right distribution of the inertia moments, etc. The second task is devoted to the selection of the actuators able to provide the necessary performances to the robotic systems. In this case, the designer needs to identify the simplest first-principle models and physical laws that make them able to effectively obtain the necessary dynamic performances as required accelerations, velocities and torques. The third task deals with the development of the electronic hardware architecture. It must be well-designed to condition and adapt signals and provide apt power supply to all the system units. In the fourth task, the basics of embedded microcontrollers are addressed along with their application in control technology. The state-of-the-art electronics technology for robotics and autonomous systems, with respect to its availability and real costs in the market, is evaluated and chosen.

The scope of the activities is to provide a basic industrially-aware knowledge for the design of small automatic systems and robotic systems that are capable to operate autonomously in indoor or outdoor environments applying lightweight control algorithms based on embedded technology.

3.2 Ballbot

In this section, the involved hardware and its setup are explained. The ballbot robot shown in Fig. 1 is designed as a test bed to investigate the issues of a dynamically stable robot balancing and driving on a single ball.

The platform was built with on-board hardware and processing. It is designed to be self-contained and to perform all the computing on board. It is relatively low cost, and the availability of its constituent parts makes it attractive for research and educational use.

3.2.1 Hardware

As shown in Fig. 1, the mechanical system is basically composed of three parts: the chassis, the propulsion system and the sphere (a basketball). The whole structure is made of alluminium and plastic in order to reduce the system's weight. The chassis is composed of three parallel

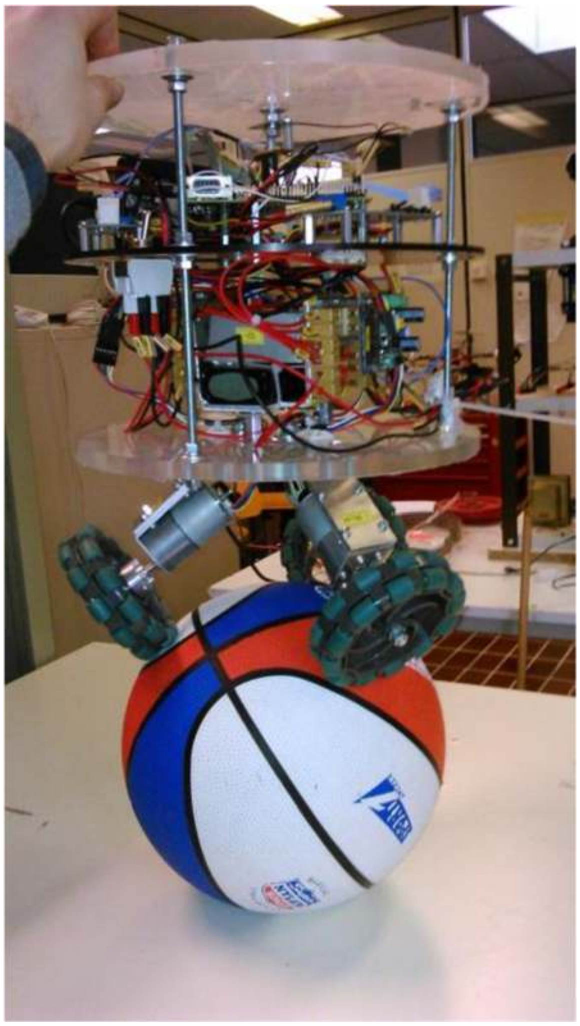
Fig. 1 The ballbot platform

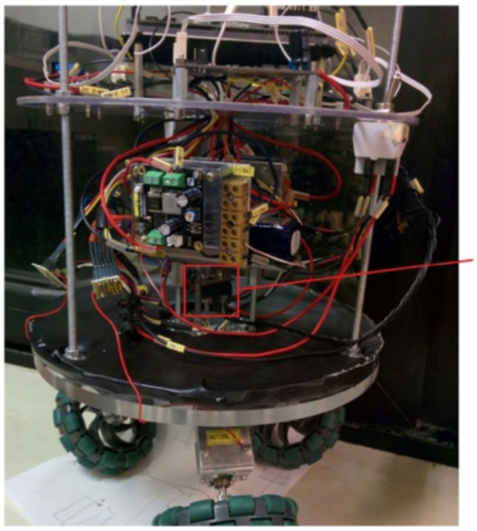

Fig. 2 The inertial measurement unit

disks having a radius of 140 mm and placed at three different levels spaced 100 mm. On the bottom of the highest level, the inertial measurement unit (IMU) can be found (see Fig. 2).

The propulsion system is composed by three DC brushed motors with gear boxes and incremental encoders, three DC-drivers and three omni wheels installed on the motor shafts. The omni wheels are a key parts of the ballbot system. An omni wheel (see Fig. 3) is a conventional wheel with a series of rollers connected to the circumference.

The axes of rotation for the rollers is parallel to the circumference of the wheel. This configuration of rollers enables the omni wheel to move in both the rotational and the lateral direction of the wheel. It can either roll around the wheel axis like a regular wheel or roll laterally using the rollers connected to the circumference or both at the same time. If actuated as normal wheel, the omni wheel allows the system to apply, in the wheel's contact point, a force which is tangential to its circumference. Moreover, it allows the system to slip sideways, if some lateral force is applied. Between the axes of the wheels, an angle β of 120° is considered (see Fig. 4). To avoid a slip motion of the platform over the ball, a minimum angle for the omniwheel contact point α of 40° is required. It is the angle between the contact point and the vertical axis of the platform (see Fig. 4).

The physical parameters describing the system are listed in Table 1.

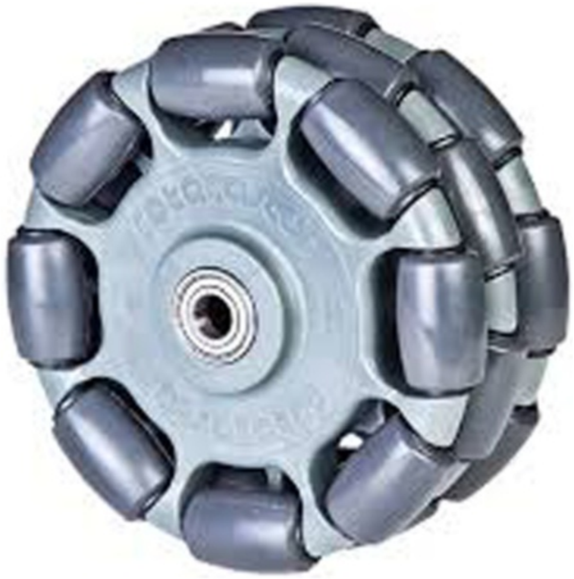
Fig. 3 The omni wheel

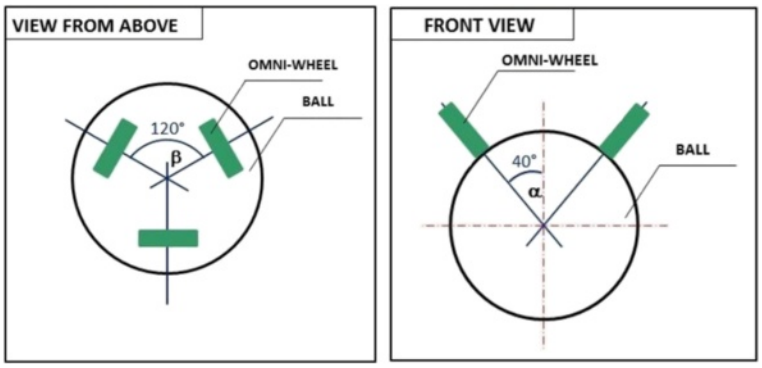

Fig. 4 Angles of the omni wheels

3.3 Robotic Leg

During the last decade, there have been significant efforts in the developing of autonomous legged robots actuated by electric motors [12, 15, 16]. Some efforts have been also done in improving the performances of the electric motors [17]. DC motors (direct current motors) are widely used in industrial systems, such as robotic manipulators, thanks to their simplicity and dependability in a wide range of operating conditions. DC motors are usually easily modeled, and linear or nonlinear control approaches can be implemented.

The powered robotic leg shown in Fig. 5 is designed as a test bed to investigate the issues of robotic legged locomotion. It is a single legged hopping that reduces the complexity of the quadrupedal (or bipedal) running, but it is still capable to explain much else about real dynamics. Furthermore, it can be useful for the evaluation of different control algorithms.

The proposed embedded system implements the accurate control of torque, speed and position of DC motors actuating the joints of the robotic leg when different control strategies are required. An important issue of these robotic systems is the ability to choose between different gaits in order to travel optimally at a certain speed or to robustly deal with perturbations. This leads to the use of different control strategies depending on the required task. Although the controller is designed for a particular robot application, it is applicable to other systems which requires to control simultaneously several DC motors with different tasks.

Table 1 Physical parameters

Parameter	Value	m.u.
Mass of the ball	0.65	*kg*
Mass of the wheel	0.107	*kg*
Mass of the body	5.045	*kg*
Radius of the ball	120	*mm*
Radius of the wheel	53	*mm*
Radius of the body	140	*mm*
Height of COG	410	*mm*
Inertia of the ball	0.005	kgm^2
Inertia of the body	0.47	kgm^2

The articulated leg consists of a main body on top of the two leg parts: the thigh and the shank. The latter is connected with a damping foot by means of a steel spring which stores the released impact energy which is then used for the upwards acceleration. The joints connecting these bodies are the knee, the hip and the joint rotating the thigh around the main body's yaw axis.

Each joint is actuated by a powerful electric DC motor for a total of three motors per leg. As showed in Fig. 5, the motors actuating the hip and the knee are installed, respectively, along the main axis of thigh and shank, whereas the actuator of the yaw axis is inside the main body. Each DC motor is a Maxon DCX35L 18V powered at 24 Vdc (its maximum allowed voltage is 30 V, a 24 V supply voltage is suited for the battery system) with a gearbox Maxon GPX42 featuring a 66:1 reduction ratio, which provides each joint with a maximum continuous joint torque of 5 Nm at 120 rpm, and a shorter duration of maximum torque of approximately 18 Nm at 65 rpm (in the latter working point the motor without a gearbox has 0.26 Nm at 7200 rpm). The motors' performance were selected to allow the leg not only to walk but also to jump. Henceforth, the basic criterion for their selection was the shorter duration of maximum torque. The range of motion for hip and knee joints are about 180^o both in flexion and in extension, while the joint of the yaw axis freely rotates 360^o.

The structure of the leg is aluminum that is reinforced and supplemented with inserts. Hip and knee joints are equipped with steel bevel gears which transmit the motion from the motor shaft to its orthogonal axis, while the joint of the yaw axis has plastic planar gears which transmit the motion from the motor shaft to its parallel axis.

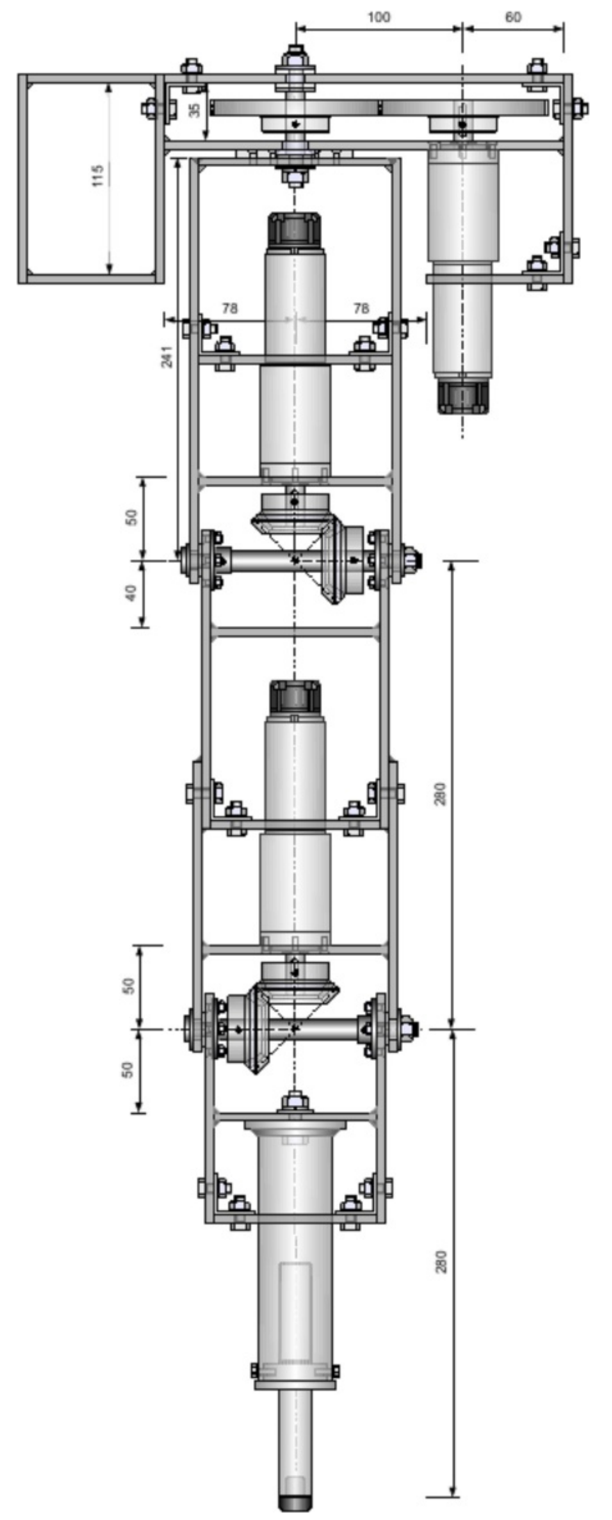

Fig. 5 Test bed of the powered robotic leg

The sensors in the leg include a rotary optical encoder for each motor to sense the joints of hip, knee and yaw axis. In addition, current sensors are used to measure the joint's torques. The maximum leg extension is about 60 cm (thigh and shank) with a total leg mass of approximatively 8 kg (the component weights, in kg, are leg structure 3.5, geared motor 0.9, bevel gear in steel 0.4, couple of plastic planar gears 0.14, spring 0.15, battery-pack 0.66). The powered leg additionally includes a custom embedded control system, whose components are located in the main body on top. The devices of the embedded control system are the MCU board, the DC motor drivers and DC/DC converters.

Figure 6 shows the schematic model of the considered robotic leg by representing its three rigid bodies (the main body on top of the leg, the thigh and the shank) and their main parameters.

The thigh is attached to the main body by two revolute joints, one base on top and one at the hip, whereas the shank is linked to the thigh by a revolute joint at the knee. The robotic leg is able to move in the three-dimensional world described by an inertial frame centered in the axes origin by means of the DC motors on each joint (see Fig. 5). This way the main body performs a spin γ around the yaw axis, the thigh performs a spin α on the hip and the shank S performs a spin β on knee. This rigid body system has consequently five degrees-of-freedom.

3.3.1 Hardware

The hardware of the robotic leg allows to power all its electromechanical and electronic parts and to communicate with all sensors and actuators installed. All electronic components have been chosen to fit comfortably into the structure of the robot. No external computers are used. In addition, the weight of all hardware modules have been chosen as low as possible. The proposed architecture is shown in Fig. 7.

The power source of the whole system is a Li-Po battery 5000 mAh 5S 65C 18.5 V providing a voltage of +12 Vdc, it powers the other devices through a power-layer of two DC/DC-converters (240 W, 24 V, 10 A) which linearly regulate and raise the voltage level to +24 V (the voltage supply of the DC motors). The switching regulators for efficient conversion (not shown in Fig. 7) provide intermediate levels of +3.3 V and +5 V to embedded systems and sensors. The DC motors are driven by DC-drivers MD10C V2 by Cytron. A first DC/DC-converter is devoted to powering the DC-driver of the hip joint (the joint with the highest load), while the second is devoted to powering the other two DC-drivers of the two other joints, respectively. The current sensor for each motor is an ACS758 by Allegro Microsystem. They are in serial connection between the DC-driver and the motor. Finally, the optical encoders HEDL-5540 by Avago Technologies are mechanically connected to the motor shafts. Figure 7 also shows the connections between the embedded control board and the other devices.

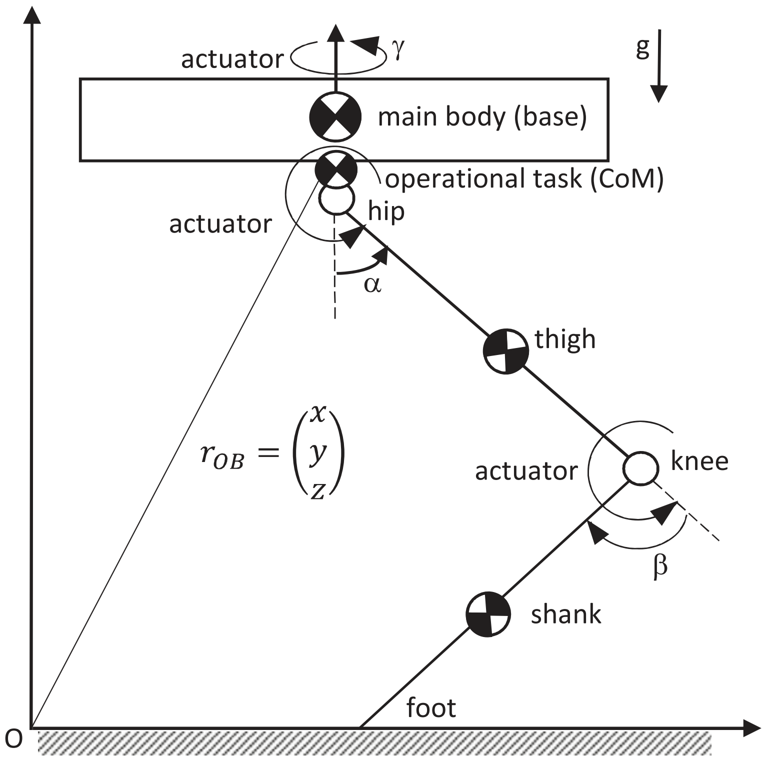

Fig. 6 Model of the robotic leg

3.4 Quadrotor

The aerial vehicle used in this work is an electric quadrotor. It is a rotary wing aircraft where each motor is attached to a rigid cross frame as shown in Fig. 8. Quadrotors are mechanically simpler than classical helicopters. Quadrotor is a vertical takeoff and landing vehicle (VTOL) able to move omnidirectionally with the ability to fly in a stationary way.

A quadrotor is controlled by varying the angular speed of each rotor. The force produced by each motor is proportional to the square of its angular speed. The front and rear motors rotate counter-clockwise, while the other two motors, the right and the left, rotate clockwise, canceling gyroscopic effects and aerodynamic torques in stationary trimmed flight.

The vertical motion is controlled by the collective thrust input, i.e. the sum of the thrusts of the motors. The forward/backward motion is achieved by a tilt around the y-axis (the axis of the left-right motors) generating a pitch angle. This is obtained controlling the differential speed of the front and rear motors. The left/right motion of the vehicle is achieved by controlling the differential speed of the right and left motors, tilting around the x-axis (the axis of the front-rear motors) and generating a roll angle. In the following, the involved hardware and its setup are explained. The quadrotor shown in Fig. 8 is designed as a test bed to investigate the issues of application of the modern control theory on embedded systems of unmanned aerial vehicles (UAVs). Also, this platform was built with on-board hardware and processing, it is designed to be self-contained and to perform all the computing on board. Its low cost and the availability of its constituent parts makes it attractive for research, educational and commercial use.

3.4.1 Hardware architecture

The quadrotor test-bed, as shown in Fig. 8, comprises light-weight alluminium rods of length 666 mm connected to a glass fiber center piece, forming the desired cross structure for the frame. Its total weight is 0.415 kg. Four motors with propellers are installed at the ends of the rods. The hardware devices are installed in the center of the frame for balancing the inertia moments. The four propulsion units are made of brushless DC motors connected to electronic speed controllers (ESC) to provide their actuation. Each motor is a Turnigy D3536/9 910 KV, a three-phase outrunner and sensorless brushless DC motor. The motor velocity constant Kv is 10 rpm/V, it provides 212 round per seconds when supplied in the range 14.8–15 V, the motor's maximum current is 21 A with a power consumption of 370 W at 15 V. Each motor has a weight of 0.102 kg and is actuated by an ESC Turnigy PLUSH-30A having a weight of 0.0425 kg. A two-blade propeller Slow-Fly-Electric-Prop 9057SF

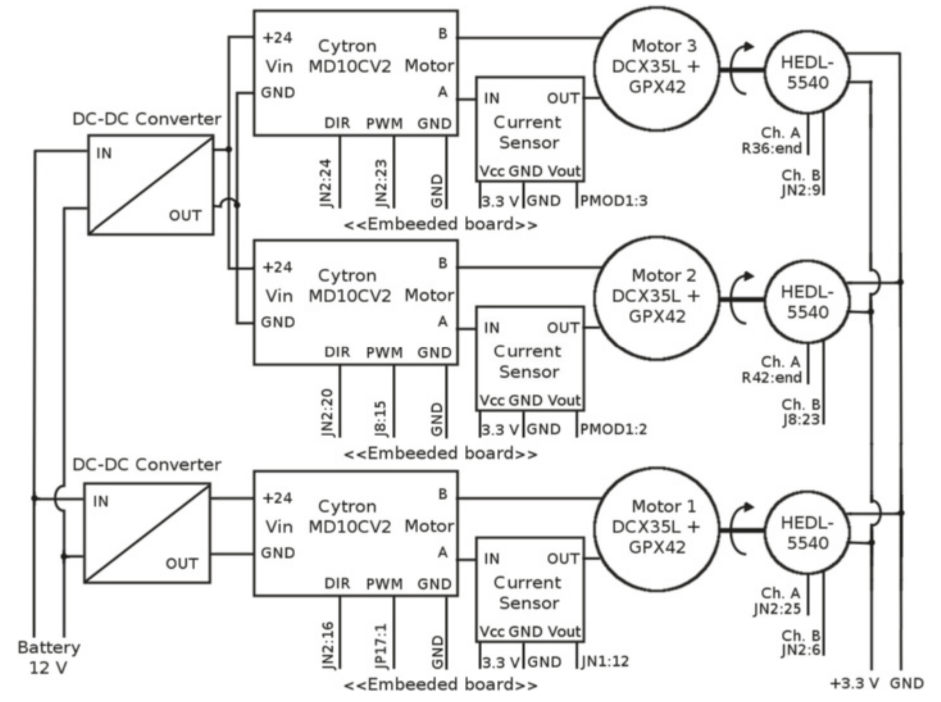

Fig. 7 Robotic leg hardware

having size 9 x 4.7 inches (9 in diameter size and 4.7 in pitch size) is directly connected to the motor's shaft, it provides a total thrust of 0.632 kg at full speed for each motor. The system is powered by a 2600 mAh, 14.8-Volt lithium polymer battery, it is a Turnigy nano-tech 2600 mAh-4S 25-50 Li-Po Pack having a weight of 0.210 kg. A sonar sensor is installed under the center of the frame for altitude measurements. In the center of the frame is found the inertial measurement unit (IMU): Drotek 10DOF V2 with a MPU6050 gyro and accelerometer plus HMC5883L magnetometer and MS5611 altimeter. The IMU has a MPU6050 module with a triple axis 3 g accelerometer (BMA180) and a triple-axis gyroscope (ITG3200), a triple-axis magnetometer HMC5883L, and an altimeter with a baromether MS5611-01BA03. The bandwidths of all the sensors are trimmed to 10 Hz. The MEMS inertial sensors

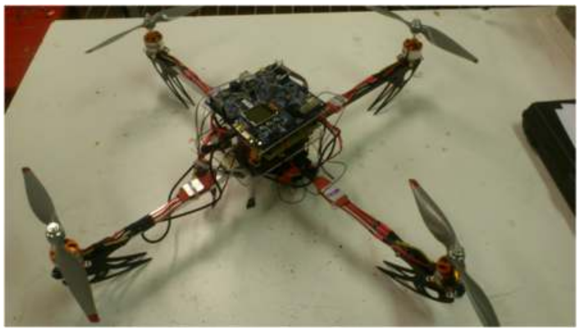

Fig. 8 Quadrotor assembly

are combined with filtering techniques for the online estimation of the attitude angles.

The hardware architecture of the quadrotor allows to power its electromechanical and electronic parts and to communicate with sensors and actuators. The platform is designed to perform sensing and computations on board. The off-the-shelf electronic components have been chosen to fit into the structure of the system. The proposed architecture is shown in Fig. 9. It shows the connections at the signal level among the embedded board and the other devices. The power source of the whole system is the Li-Po battery which provides a voltage of +14.8 Vdc, which powers the electronic-power-board that linearly regulate the voltage level and distribute it to the ESCs by means of a power-divider hardware. Furthermore, the electronic-power-board decreases the voltage level to both +5 and +3.3 V needed to power the sensors and the embedded boards. Figure 8 shows hardware layout and points out the electronic and electromechanic parts of the quadrotor.

4 Methods

A suitable hardware and software structure has been identified as a common ground to satisfy the needs of our three experimental robots. The chosen embedded system consists of an evaluation board with its several modules and components. The evaluation board is only adopted for prototyping purposes, while the final redesign of a customized board has to be foreseen for a final optimized

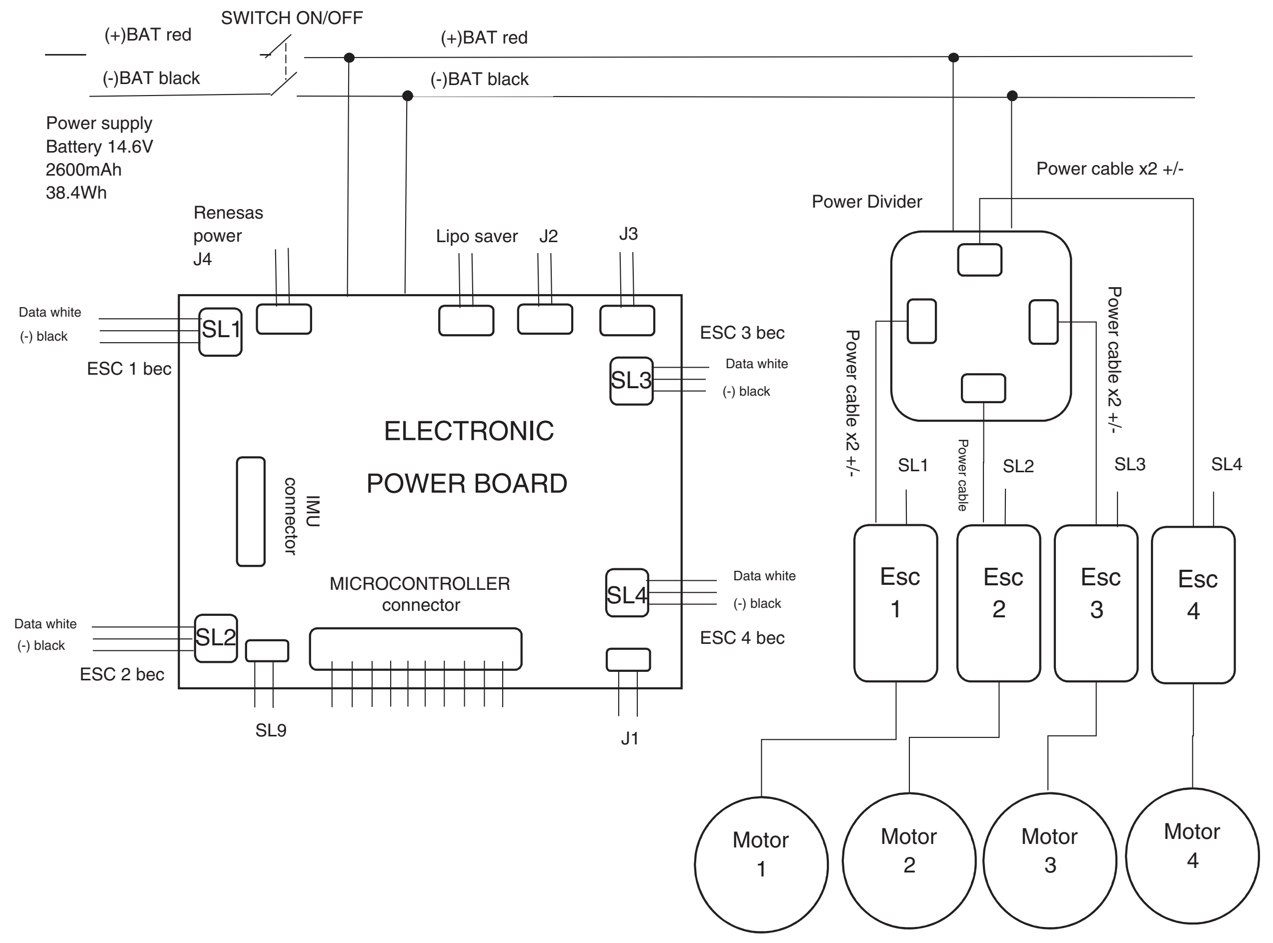

Fig. 9 Hardware schematic of the quadrotor

prototype with unused components pruned and so ready for real series production.

Embedded software has been implemented, over the environment of the compiler and driver libraries available from the MCU vendor, in order to enable the communications between the different hardware devices of the robot, namely three DC motors with gearboxes and angular encoders, on inertial measurement unit (IMU) and the robot's controllers. Our software provides the necessary filtering of inertial measurements and the communication with external environment.

With the help of the proposed embedded system architecture, each motor can selectively operate in torque or speed or position control mode, and this can be done for each of three DC motors. In particular, the hardware of the robotic leg is composed by three motors with a gearbox. Each motor is equipped with a rotary optical encoder and with a low-cost current sensor. The hardware of the embedded system is based on a commercial low-cost embedded board equipped with a 32-bit micro controller unit (MCU). The embedded software implements the control by using only the peripherals available on board. A serial communication protocol is used to send the sensor signals to a MATLAB-Simulink™script running on a PC which stores and shows the system behaviors through a serial communication connection. The performance of the proposed embedded control system is assessed by experiments on the motor and by simulation tests on the motor's mathematical model.

4.1 Embedded hardware choice and holonic software structure

The choice of the embedded architecture both for hardware and software provisions has been oriented to a real market replication of the robotic solutions here studied. Although many choices are currently available for implementations, here, we are not only concerning with scientific results but also with the technological issues and costs of the solutions. The current choice of hardware components and boards has been influenced from the products that are actually available on the market, at least for the next 5 years. Of course, we do not rely completely on current products but we suggest some enduring design criteria for the setup of the architecture choice basing on the case studies at hand. Some practical guidelines should be followed to let scientific experiments be oriented to a fast market take-up of the solutions by assessing their feasibility, cost and scalability of the technologies involved. When we chose a hardware architecture for the experiments, we have to bear in mind that it should be produced and engineered with the best available technology.

Having the robotic solutions here illustrated conceived in didactic context, the aim is the instillation of the awareness in robotics students and communities about the engineering requirements and design constraints in typical implementation of the interested class of robots. The current choice went toward an interesting set of commercially available micro-controller unit (MCU) solutions provided by Renesas. Attention was paid to the motor control oriented features of the RX series microcontrollers.

With finding most of the needed peripheral and device provisions available on-chip, it guarantees a good design starting point for a customized board at reduced costs and ready for mass production. Moreover, we had in mind a clear and scalable hierarchy of parts that must add-up to constitute a more complex robotic system featuring artificial and distributed intelligence, possibly in cloud systems [18].

The main underlying idea is to pursue a holonic approach [19] where the regulatory layer can be put in connection, though semantically separated, to the higher intelligent, collaborative and distributed layer. So, the choice of RX series MCU has been considered ideal for the regulatory layer where the control policy can be implemented while in connection to a wider and more complex system, by the use of apt design of the low-level control scheduling policies. Holonic and collaborative system configuration is achievable within a real-time operating system infrastructure, which can handle the low-level regulation while managing the higher information interconnection.

The first choice went to the RX63N, RX631 group which consists of 32-bit microcontrollers capable of operating at up to 100 MHz with on-chip FPU, DSP instruction set, with nominal 165 Dhrystone MIPS computing performance. Of course, they can not be claimed as the fastest nor the best MCUs, but already stable and in mass-production. Moreover, our practical exercise is to put ourselves within the typical design constraints encountered in embedded electronics engineering, not a comparison among MCUs architectures and producers. The chosen constraints of course should guide the designer to an optimal solution just enough to address the needs, and this is the good practice exercise here suggested.

RX63N series feature up to 2 Mbytes fast coupled internal flash memory, that can allow the fast execution-in-place of a low footprint real-time operating system and all the needed control and communication algorithms. It provides up to 256 kbytes of static RAM and added 32 kbytes of data-specific flash for fast data access and DSP computing.

The RX63N has an Ethernet controller with dedicated direct memory access (DMA), which is suitable for networking (in holonic sense) and a hardware communication encryption unit (i.p. AES encryption and decryption functions) that is a must-have in all the devices approaching the Internet of Things. This MCU family is mostly interesting for their plenty of timers that can be used in motor control both for sensing and actuation, in particular, the multi-function timer pulse unit (MTU) designed for motor control and PWM generation and sensing. MTU is a multi-purpose timer peripheral that allows the sensing and output of a vast set of waveforms that are essentials in motor control issues.

Given the examples discussed here in this paper, a commercial hardware board that implements the regulation and control part of the one of the three systems can be safely attainable with less than US$20 in mass production, lest special quality production needs that may raise the bill of materials cost. So, we consider our experiment framework an interesting guideline and test-bed for didactic and commercial solutions in self-balancing robotics components and applications.

4.2 Real-time scheduling

Real-time scheduling of concurrent processes and actions is a hot theme in embedded systems [20]. In our robotics case, the regulation layer must act within hard real-time constraints as the controllers can not miss any contact with the physical world dynamics. At the same time, a collaborative and holonic structure needs be in timely communication with the higher control layers where real-time constraints can not be guaranteed or not strictly needed. A currently common solution, which many practitioners adopt, is the deployment of small-footprint real-time operating systems and the strict separation among deferrable and preemptible processes and uninterruptible ones.

The treatment of an optimal use of CPU resources under advanced scheduling policy is beyond the scope of this work. In the present experiments, things have been kept as simple as possible to rely on good hardware parallel architecture capabilities for the specific and limited number of actions in the control layer.

In most of the self-balancing systems, the control is obtained with a measurable duration of the computation, action and acquisition. By measuring the duration of a control period, the sequence of discrete-time controls on temporal basis can be easily scheduled. In these simple cases, a round-robin policy is enough to achieve the results. A simple scheduling solution should rely on the peripheral provisions of the RX architecture. In particular, scheduling events can be controlled by the periodic interrupts generated with the programmable compare match timer (CMT) unit. These interrupts are used to schedule among different processes and uninterruptible driver phases during the control period. After a measuring of these deterministic periods and appropriate time guards, the remaining time slice in the control period is to be reserved to external remote interaction and to the communication of parameters, information exchange and interaction with other machines. Modern lightweight RESTful approaches and similar approaches should be allowed also on tiny and resource constrained devices [21].

The processes involved in external communication, and at the highest level in holonic architecture, must be preemptible and interruptible, so their priority is kept lower and they can be scheduled on a session basis. Indeed most of them are prone to a packet-wise communication like in UDP/IP-based connections.

The rest of the control action, on the contrary, has to act and react with hard real-time limits. The duration of the acquisition, processing and output transduction must be fixed and deterministic. Within those design constraints, a 150-ms control period is chosen. It has been assessed as a suitable control period for the three applications at hand, basing on robust control theory and experience in the field. During this computing period, four motors are controlled, the sensing and control processing is performed and then a time slot of nearly 80 ms is left for the higher level tasks, like UDP/IP-based communications. Being this last task preemptible and interruptible, it is allowed to be spread across several control periods to complete with no harm to the overall application purposes.

The use of a more effective and advanced real-time scheduling technique, as the earliest deadline first (EDF), would allow for an optimal responsiveness of the system at possibly higher control rates. Nevertheless, the design overhead for the implementation of EDF scheduling can be avoided in simple cases if the accurate design of robust control algorithms can compensate inherently the rigidity and limitations of the round-robin scheme.

4.3 Embedded system: hardware and software

The design of accurate and efficient embedded control system is a key issue to ensure the correctness of the robotic leg's movements. Below, both hardware and software topics will be addressed. The proposed embedded hardware (Renesas YRDKRX63N) is not based on expensive controllers for single motors but on a general purpose evaluation board which are proposed for driving at the same time at least three motors of the robotic leg, of the robotic ballbot and the four motors of the quadrotor. The embedded board has a 32-bit RX63N MCU of the RX family/RX600 series. It communicates with the embedded devices and sensors.

The functional diagram in Fig. 10 shows the functional interactions between embedded sensor/devices and MCU's peripherals. Figure 10 shows that the low-level embedded control reads sensor data and generates a PWM signal for the DC-driver unit which drives the DC motor. Figure 10 also shows the board's peripherals involved in the control process. The MCU uses an on-chip compare match timer (CMT) for timing the processes, the multifunction timer unit (MTU) generates PWM signals while, MTU and timer pulse unit (TPU) are used to acquire data from encoders. The analog to digital converters 10/12 bit (ADC 10/12) are devoted to read the current sensors while general purposes input/output (GPIO) digital ports are used to define the shaft's rotation direction. The serial communication interface (SCI), set in universal asynchronous receiver-transmitter (UART) mode, is used to send the variables of the motor and of the sensors to a MATLAB-Simulink™script running on a PC for the early experimental tests.

4.4 Embedded software

In this section, the software architecture of the embedded control system is presented. The block diagram of the feedback control system for a single DC motor, which is implemented on the embedded system, is shown in Fig. 11.

The software implementing the embedded system is made of four main parts: initialization, timing and interrupt routines, acquisition of sensor measurements and controller implementation. In the following subsections, these parts will be introduced.

The reference signals, θ_r rather than ω_r or i_r are set using a high-level controller, which description is out of

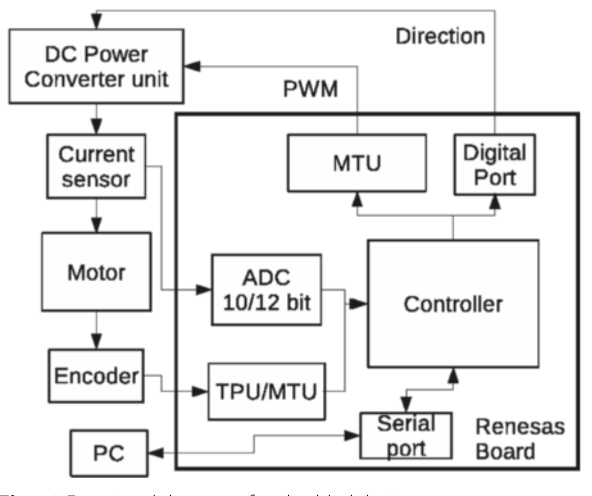

Fig. 10 Functional diagram of embedded devices

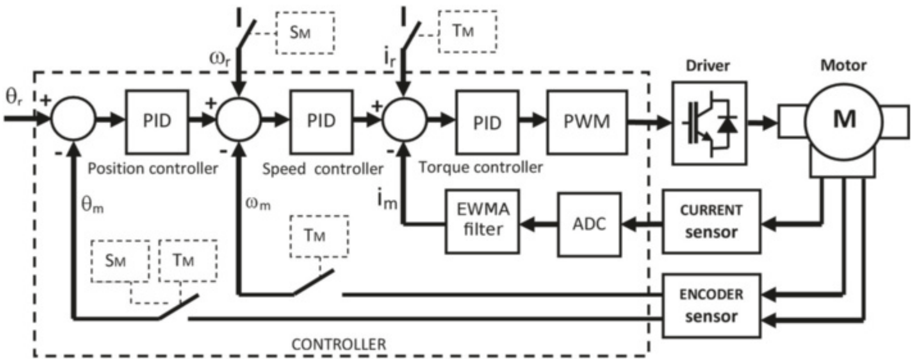

Fig. 11 Block diagram for feedback control of DC motor

the scope of this paper. T_M and S_M represent switches that select control modes: the torque mode and the speed mode, respectively. The position θ_m and speed ω_m of the motor are measured using the signals generated by the encoder. The difference between reference and measured signals are, respectively, the input for the position or the speed for the control algorithm. The position and the speed controllers set the reference for the current controller. Otherwise, it is possible to set directly the motor torque from the high-level controller. Basing on the difference between the reference and the measured current, the current controller computes the new duty cycle value of the PWM control signal. The PWM generated by the microcontroller controls the driver of the DC motor. The measured current is also used for the overcurrent protection—if the current exceeds a certain limit, the PWM control signal is disabled. In the current feedback loop, an exponentially weight moving average (EWMA) filter is used to obtain a reliable current signal from the current sensor. The software implementing the embedded control system of a single DC motor can be divided into four main parts: initialization, timing and interrupt routines, acquisition of sensor measurements and controller implementation. In the following subsections, these parts will be introduced and Fig. 12 shows, as an example, the flowchart of the embedded software when the position feedback control mode is selected by high-level controller.

4.4.1 System initialization

In this subsection, the steps required for the initialization of the system are presented. At first, all the peripherals used by the embedded board are initialized according to the hardware user's manual of the RX63N microcontroller. The used variables are also initialized. Before the start of the main loop, the voltage offset of the current sensor is evaluated to ensure the correctness of the voltage-to-current conversion. In order to remove the offset error, a preliminary cycle of 500 ms is done when no current is absorbed by the motor, so that the voltage reading will become the zero ampere reference.

4.4.2 Timing and interrupt routines

An embedded control system assures real-time constraints through the interrupts of the MCU. Synchronous interrupts are required for the control algorithm of the

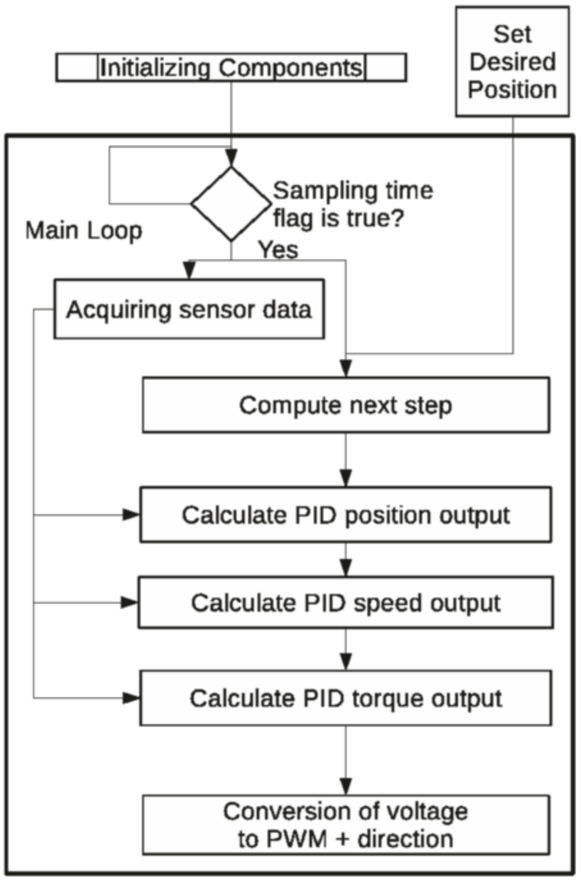
Initializing Components
Set Desired Position
Sampling time flag is true?
Main Loop
Yes
Acquiring sensor data
Compute next step
Calculate PID position output
Calculate PID speed output
Calculate PID torque output
Conversion of voltage to PWM + direction

Fig. 12 Embedded software flowchart: position control mode

DC motor. The synchronous interrupts are generated in the main loop and they are timed every 5, 10, and 80 ms by the CMT peripheral; in particular, the first timing slot is used for each data acquisition, the second timing slot is used for both data acquisition and control algorithm, and the third timing is used to manage the communication with an external device or PC. Asynchronous interrupts are instead used for the acquisition of the encoder readings. Synchronous interrupts are required for the control algorithm of the DC motor.

4.4.3 Acquisition of sensor measurements

The motor current is measured using a current sensor and is acquired using the analog to digital converter (ADC 10/12) of the embedded board. The ADC can generate high-precision results, but a filtering procedure is required due to the low resolution of the current sensor. This ADC data acquisition is timed 5 ms in order to have the sensors measures ready for the control loops but the readings of the ADC peripheral will be available only every 10 ms because we let 5 ms for data processing. The encoder readings are done by the MTU and TPU peripherals that use asynchronous interrupts. The query function to retrieve encoder's data is read at the same time of the ADC reading that is every 10 ms. The encoder generates two quadrature encoder signals. These signals are applied on two MTU (or TPU) input type pins. The encoder is configured to generate processor interrupts at a fixed rate. The speed of the motor is computed basing on the number of encoder pulses per revolution. The instruction clock cycle of the microcontroller, the number of edges which are accounted by the MTU (or TPU) of the microcontroller (rising, falling or both edges) and so the number of counted edges during the considered time interval. The value is loaded in the register of the timer associated to velocity measurement and has to be set such that the divisor is a power of two, and to perform a register shift instead of a division.

The IMU data are retrieved using an I^2C protocol provided by the board on-chip peripheral RIIC and supported by our peripheral driver that uses interrupt handling routine to drive the device through apt operations.

4.4.4 Controller implementation

For the ballbot and the robotic leg applications, three discrete time-cascaded PID controllers with anti-windup system are implemented for each motor [22, 23]. The computations are performed by the same function for each controller using different parameters. The cascaded controllers work according to the scheme of Fig. 11 and generate the input voltage for the DC motor. The last step of the control algorithm, shown in the flowchart of Fig. 12, is to convert the voltage value in a PWM signal and generate a direction signal in order to drive the DC driver. The PWM signal is generated by the MTU peripheral, where a function is called to convert the voltage value into the register counter which defines the PWM signal. The controller will generate the input voltage for the DC motor. The last step of the control algorithm is to convert the voltage value in a PWM signal and to generate a direction signal in order to drive the DC driver. The PWM signal will be generated by the MTU peripheral.

A linear full state-feedback LQR (linear quadratic regulator) controller based on a linearized model can be proposed for the system control, in particular, in the ballbot case. The computations can be performed by the same embedded board.

For the quadrotor, the former PIDs are not needed because the control of the rotary wings is performed completely by the ESC units.

5 Experimental results

In this section, we selected two of the proposed systems as experimental tests: the ballbot system and the robotic leg. The proposed design of an embedded system for a ballbot robot is a starting and reference point for educational and research uses. It is used to make the users able to design an embedded cyber-physical system in all of its parts, starting from the high-level programming of a microcontroller and its low-level-specific coding approach like the writing of peripheral driver according to real-time and communication protocol constraints. In the following, the focus is on the most relevant aspects which are common in these kind of systems. In particular, focus is on the tilt sensing, the step input motor response, and the motor response along different tracks. The tilt sensing is performed through the IMU sensor which is the fundamental component for the self-balancing capability. The motor response to a step input shows the dynamic profile of the actuation that is relevant to the calibration of the motor controllers. Then, motor response in different tasks show the performance of the controlled actuators and of the control system. The first experimental results are described in Fig. 13 where there are represented plots of the roll angle values sensed by the IMU, filtered with a Kalman filtering implemented on board and sent to the PC via UART protocol using MATLAB-Simulink™. As shown in Fig. 13, the Kalman filtering smooths the roll signal profile and reduces the system's sensitivity to the higher frequency oscillations that are usually related to sensing noise or contol transients. The performances are shown in the roll error plot where the difference between the filtered and non-filtered signals are compared. Figure 14 shows the Motor 1 velocity profile used to identify experimentally the motor model, acquired via UART as before. In that figure, the motor angular velocity shows a transient response of nearly 100 ms after which it reaches the steady state. The observed oscillations in the steady state are

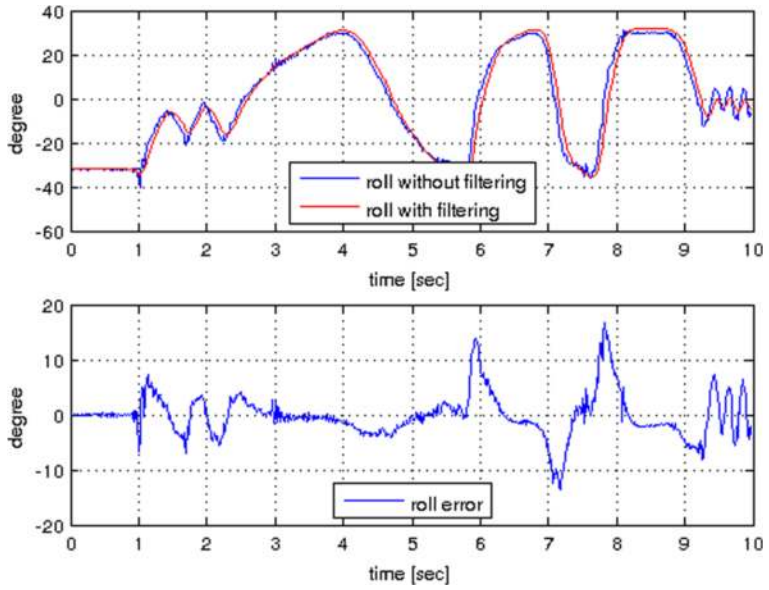

Fig. 13 Roll sensor measurement

related to the motor encoder resolution and are not relevant for system's control purposes. During self-balancing tasks, the motor works mainly with transient response due to the high system instability.

The proposed embedded system has been tested on all the underactuated systems. For brevity of exposition, the tests on the DC motor of the robotic leg are shown and compared with the simulation results. The performance in terms of tracking capability has been evaluated. The simulation results of the proposed controllers are shown in Fig. 15. It shows, respectively, the details of position, speed and torque tracking after set-point changes toward the desired angular position, which switches between 0° and 180°. Figure 15 shows that the proposed controller is able to accurately track the position changes, the speed and torque that are concurring to the same tasks. In the same figure, the effect of disturbance load during the control task can also be appreciated. At time intervals of 10 s, a load torque is applied to the rotor shaft. It is worth noting that the controllers are able to perform the required task and to react to the disturbance. The experimental results of the embedded controllers tested on the real DC motor for the same task are shown in Fig. 16. It shows, respectively, the details of position, speed and torque-tracking acquired values after set-point changes on desired angular position, which switches between 0° and 180°. Figure 17 shows the evolution of the position error over time between the desired position and the real position during the experimental tests. The figure shows the good performance of the motor controller in relation to the aforementioned results of previous figures. In particular, the last figure shows an appreciable position error only during reference changes and negligible error elsewhere.

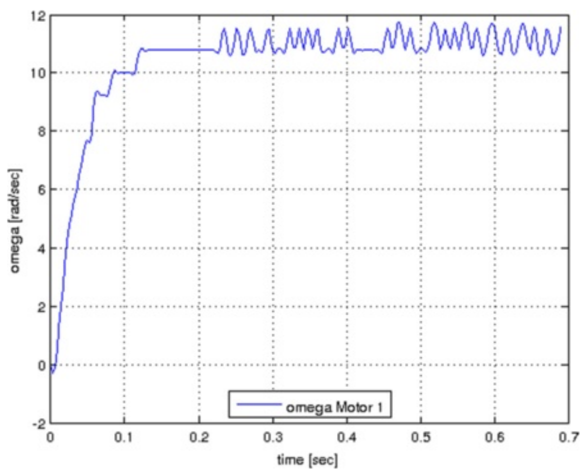

Fig. 14 Motor 1 angular velocity profile

6 Conclusions

In this paper, we propose and present a design framework for embedded electronics systems conceived to satisfy a class of highly unstable, underactuated and self-balancing robotic systems. We describe some design guidelines that are suggested to the effective and low-cost realization of the devices taking into account a commercial perspective of the products attainable from the prototypes under test. The methodology has been validated on the three case studies of a ballbot and a legged robot along with

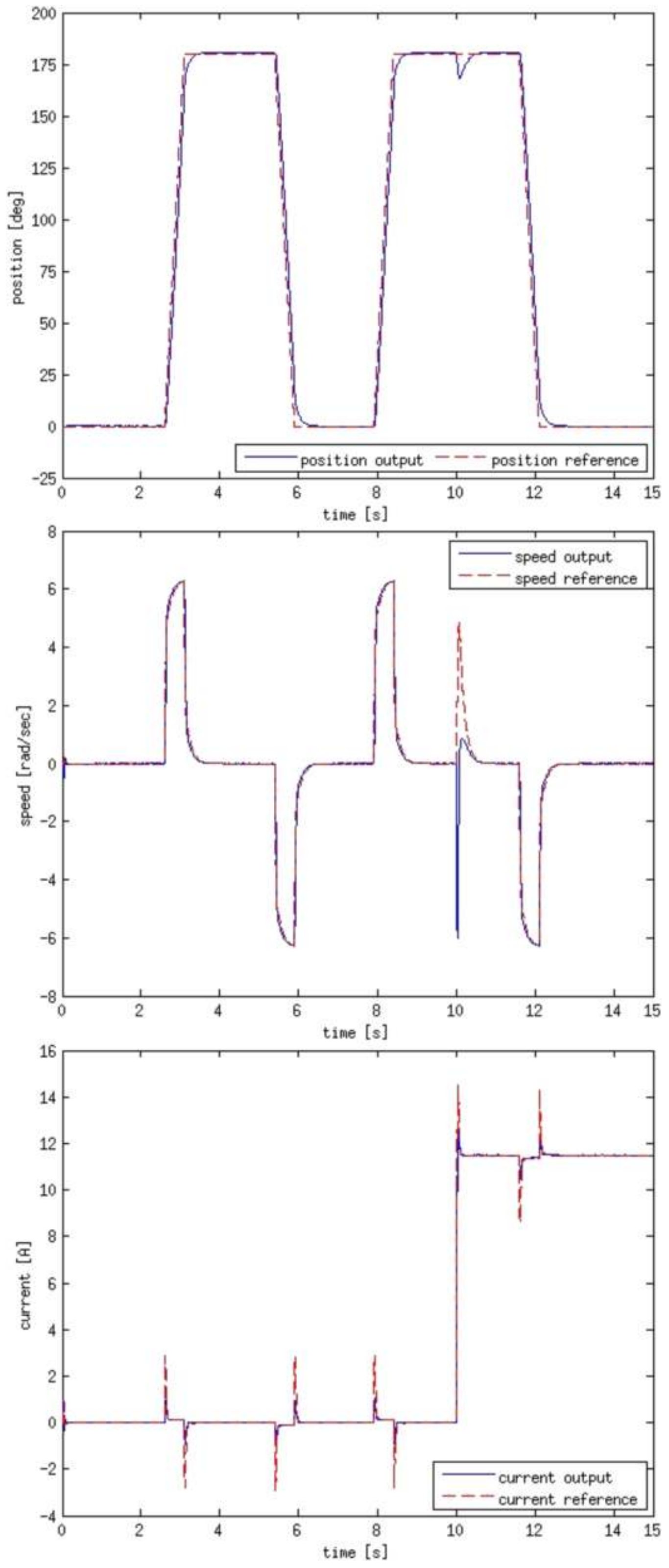

Fig. 15 Simulation results

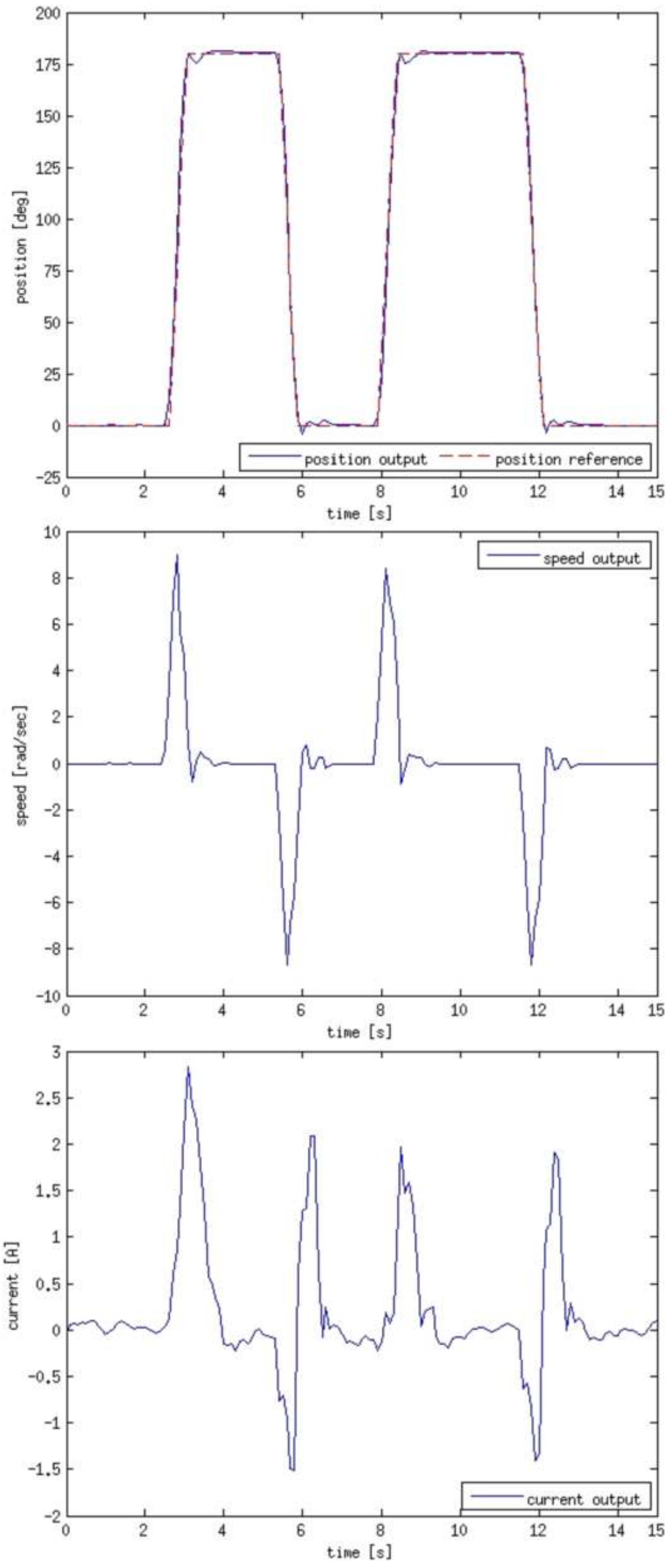

Fig. 16 Experimental results

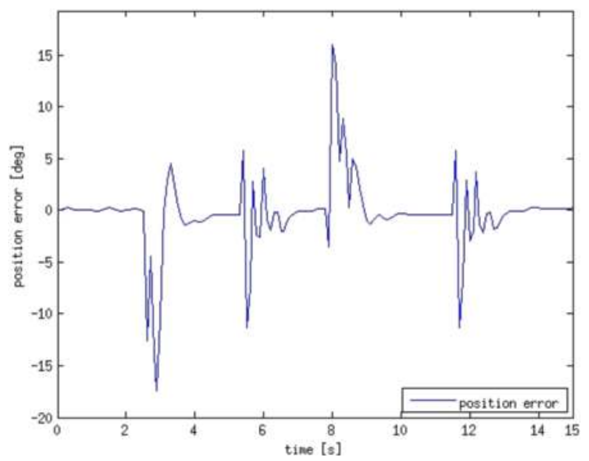

Fig. 17 Position error in experimental test

a quadrotor application. The work is born in a didactic context where students are trained to front the real-world engineering design problems where final production costs represent a major driver for the embedded electronics solutions. By following the same methodology, which pays attention to the hardware architecture provisions of MCU vendors and to the overall cost of the components, developers are guided toward gaining awareness of a best-effort approach viable for the large-scale commercialization of the solutions. Future work will address a comparative validation on different MCU vendors for the class of problems at hand.

Moreover, the design issues considered in the real-time aspects of the embedded programming allow the solutions to be compliant with the wider class of cyber-physical systems and at the same time, to implement the software hierarchy and structure needed from the holonic concept in collaborative mechatronics and industrial applications.

The design of an embedded system for a ballbot robot for research and educational purposes has been proposed. The efforts to develop a low-cost integrated system have been explained. Hints for an embedded control system are proposed to cope with the management of the whole system. The feasibility of embedded control in terms of computational power should be demonstrated through real tests on the embedded microcontroller. Future work includes the application of a linear control technique for stabilization purposes. Furthermore, the development of a high-level controller will be considered to assess the generation of the trajectories of the ballbot. Tests on the ballbot prototype are currently under investigation.

Concerning the legged robot, an embedded low-level control of position, speed and torque for DC motors devoted to actuate joints of a robotic leg is proposed. An embedded control system is used to cope with the low-level control of three DC motors. The feasibility of embedded control in terms of required computational efforts has been demonstrated through real tests on a low power embedded microcontroller. Experimental results show that the embedded control can provide the desired performances for the robotic leg. Future work includes extending the proposed scheme to different control techniques.

The quadrotor assembly is interesting and completes the framework, as the embedded platform has been pushed to control the four motors needed by the application. Also, in this case, future work will involve high-level collaborative control and low-power issues.

Future work on the framework will extend its validation to include other aerial vehicles as ducted fan aerial robots, currently under development.

Competing interests

The authors declare that they have no competing interests.

Received: 11 March 2016 Accepted: 23 July 2016
Published online: 09 August 2016

References

1. A Bonci, S Imbrescia, M Pirani, P Ratini, Rapid prototyping of open source ordinary differential equations solver in distributed embedded control application. IEEE/ASME 10th International Conference on Mechatronic and Embedded Systems and Applications (MESA 2014), Senigallia, Italy, 1–6 (2014)
2. A Rajhans, A Bhave, I Ruchkin, BH Krogh, D Garlan, A Platzer, B Schmerl, Supporting heterogeneity in cyber-physical systems architectures. IEEE Trans. Autom. Control. **59**(12), 3178–3193 (2014)
3. U Nagarajan, A Mampetta, GA Kantor, RL Hollis, State transition, balancing, station keeping, and Yaw Control for a dynamically stable single spherical wheel mobile robot. IEEE International Conference on Robotics and Automation ICRA'09, Kobe, Japan, 998–1003 (2009)
4. TB Lauwers, GA Kantor, RL Hollis, A Dynamically stable single-wheeled mobile robot with inverse mouse-ball drive. IEEE International Conference on Robotics and Automation ICRA'06, Orlando, Florida, USA, 2884–2889 (2006)
5. M Kumagai, T Ochiai, Development of a robot balancing on a ball. IEEE International Conference on Control Automation and Systems ICCASŠ08, Seul, Korea, 433–438 (2008)
6. J Fong, S Uppill, in *Report*. Design and build a ballbot (The University of Adelaide, Australia, 2009)
7. S Leutenegger, F Colas, P Fankhauser, C Gwerder, Modeling and control of a ballbot. Bachelor Thesis, ETH Zurich (2010)
8. Santos Gonzales de PG, E Garcia, J Estremera, *Stability in walking robots. Quadrupedal Locomotion. An Introduction to the Control of Four-legged Robots*, vol. 1. (Springer, Berlin, 2006), pp. 33–54
9. MH Raibert, *Legged robots that balance*. (MIT Press, Cambridge, Mass., 1986)
10. SM Song, KJ Waldron. Machines that walk: the adaptive suspension vehicle (The MIT Press, Cambridge, MA, USA, 1988)
11. C Semini, NG Tsagarakis, E Guglielmino, M Focchi, F Cannella, DG Caldwell, Design of HyQ a hydraulically and electrically actuated quadruped robot. Proc IME Part I: J. Syst. Control Eng. **225**(6), 831–849 (2011)
12. M Hutter, C Gehring, M Bloesch, MA Hoepflinger, CD Remy, R Siegwart, in *Proceedings of the 15th International Conference on Climbing and Walking Robots (CLAWAR)*. StarlETH: A compliant quadrupedal robot for fast, efficient, and versatile locomotion, (Illinois, USA, 2012)
13. R Lozano, *Unmanned aerial vehicles: embedded control*. (John Wiley & Sons, 2013)

14. J Escareno, S Salazar-Cruz, R Lozano, in *Proceedings of the American Control Conference ACC'06,*. Embedded control of a four-rotor UAV , (Minneapolis, Minnesota, USA, 2006)
15. M Azad, R Featherstone, Balancing and hopping motion of a planar hopper with one actuator. IEEE International Conference on Robotics and Automation (ICRA 2013), Karlsruhe, Germany, 2027–2032 (2013)
16. MP Murphy, A Saunders, C Moreira, AA Rizzi, M Raibert, The LittleDog robot. Int. J. Robot. Res. **30**(2), 145–149 (2010)
17. N Paine, L Sentis, A new prismatic series elastic actuator with compact size and high performance. Paper presented at the IEEE International Conference on Robotics and Biomimetics (ROBIO), Guangzhou, China (2012)
18. L Wang, M Liu, MQH Meng, Real-time multisensor data retrieval for cloud robotic systems. IEEE Trans. Autom. Sci. Eng. **12**(2), 507–518 (2015). doi:10.1109/TASE.2015.2408634
19. J Barbosa, P Leitão, E Adam, D Trentesaux, Dynamic self-organization in holonic multi-agent manufacturing systems: the ADACOR evolution. Comput. Ind. **66**, 99–111 (2015)
20. A Biondi, GC Buttazzo, M Bertogna, Schedulability analysis of hierarchical real-time systems under shared resources. IEEE Trans. Comput (2015). doi:10.1109/TC.2015.2444833
21. E Latronico, EA Lee, M Lohstroh, C Shaver, A Wasicek, M Weber, A Vision of swarmlets. IEEE Internet Comput. **19**(2), 20–28 (2015). doi:10.1109/MIC.2015.17
22. R Dorf, R Bishop, *Modern Control Systems*, Ch. 4. (Prentice-Hall, Englewood Cliffs, 2001), pp. 174–223
23. JG Ziegler, NB Nichols, Optimum settings for automatic controllers. ASME J. Dyn. Syst. Meas. Control. **115**(2B), 220–222 (1993)

Submit your manuscript to a SpringerOpen journal and benefit from:

► Convenient online submission
► Rigorous peer review
► Immediate publication on acceptance
► Open access: articles freely available online
► High visibility within the field
► Retaining the copyright to your article

Submit your next manuscript at ► springeropen.com