Article

PiBot: an open low-cost robotic platform with camera for STEM education

Julio Vega [1,†,‡], **José M. Cañas** [1,‡]

[1] Rey Juan Carlos University; julio.vega@urjc.es
[2] Rey Juan Carlos University; jmplaza@gsyc.es
* Correspondence: julio.vega@urjc.es; Tel.: +34-914-888-755
† Correspondence concerning this article should be addressed to Julio Vega, Department of Telematic Systems and Computation, Rey Juan Carlos University, Camino del Molino S/N, 28934 Fuenlabrada, Madrid, Spain.
‡ These authors contributed equally to this work.

Abstract: This paper presents the robotic platform, `PiBot`, that has been developed and that is aimed at improving the teaching of Robotics with vision to secondary students. Its computational core is the Raspberry Pi 3 controller board, and the greatest novelty of this prototype is the support developed for the powerful camera mounted on board, the `PiCamera`. An open software infrastructure written in Python language was implemented so that the student may use this camera, or even a WebCam, as the main sensor of this robotic platform. Also, higher level commands have been provided to enhance the learning outcome for beginners. In addition, a `PiBot` 3D printable model and the counterpart for the Gazebo simulator were also developed and fully supported. They are publicly available so that students and educational centers that do not have the physical robot or can not afford the costs of these, can nevertheless practice and learn or teach Robotics using these open platforms: `DIY-PiBot` and/or `simulated-PiBot`.

Keywords: Teaching Robotics; Science teaching; STEM; robotic tool; Python; Raspberry Pi; PiCamera; vision system

---

1. Introduction

The appearance of robotic devices in the mass market such as robotic vacuum cleaners and mops, as well as numerous applications and existing domotic services have made this technology increasingly present in the daily routine of society, not to mention other frequently automated tasks: withdrawing money at the ATM, automatic payment in supermarkets, or the massive use of Internet, shopping, banking, and much more.

Furthermore, autonomous cars or drones make the use of this technology more visible and reinforce its appeal. In fact, the short and mid-term future is/will be marked by industrial production dominated by intelligent machines ([1]). The presence of humans in these *intelligent factories* tends to be increasingly reduced and will eventually be symbolic and sporadic.

There is no doubt that a machine's capacity for taking optimum decisions in real time and simultaneously handling an enormous quantity of data, is far greater than that of a human being. The so-called *Industrialization 4.0* ([2]) involves the integration of complex robotic systems in factories (Figure 1 right), logistics and what is known as the *Internet of things*, where sophisticated automatons handle an immense quantity of data to take strategic decisions for companies.

These mobile and intelligent robots need, in addition to a large computational capacity, a complex sensory system to *act* intelligently not only in factories but in robot-human interaction at general level ([3]). The fixed automation of structured production chains is giving way to an unpredictable world

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 17 October 2018 doi:10.20944/preprints201810.0372.v1

Peer-reviewed version available at *Electronics* **2018**, *7*, 430; doi:10.3390/electronics7120430

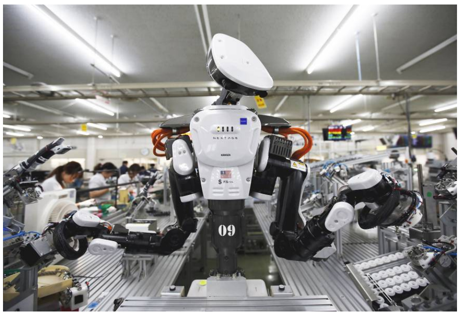

Figure 1. 4th Industrial revolution: Intelligent robots at Glory Ltd.

and a totally unstructured *reality* which makes evident the need for a wide complementary range of sensors and actuators to attain complete autonomy ([4]).

Although visual sensory modality has not been the most used for some years in mobile robotics (sonar and/or laser have been more used as sensors), at present it has become the most widely used sensor and will definitely be the most commonly used in the long-term future, because of the possibilities it offers and the power of calculation of current computers. They are low-cost devices which are potentially very computationally rich, since they provide a lot of information.

However, visual capacity in robots, in contrast to that of living beings, is not an easy technique. The main difficulty lies in extracting useful information from the large amount of data that a camera provides, for which good algorithms are needed.

Summarizing, as described, the advance of Artificial Intelligence (AI), Robotics and automation in society ([5]), the future of work and industry in particular converge in what is already mentioned as the fourth industrial revolution. According to the analysis of the University of Oxford ([6]) and the professional services of Deloitte ([7]), almost half of all jobs will be occupied by robots in the next 25 years. Furthermore, as the Mckinsey institute shows in its last report on the global economy ([8]), robots will perform the work of about 800 million jobs in 2030.

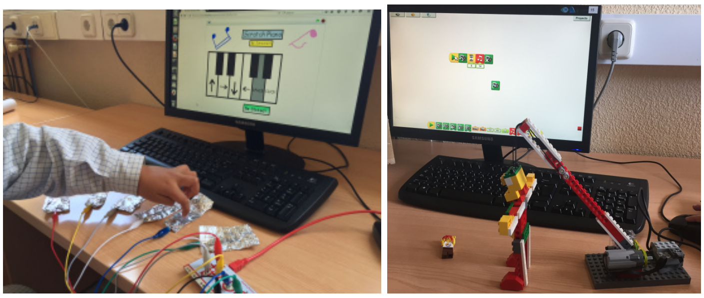

Figure 2. Different robotic prototypes to work in different educational areas

It is therefore of vital importance to incorporate technology, and specifically Robotics with vision systems, in the pre-university educational system since todays' youngest students will be those who, within a decade, have to confront a labour market that will demand profiles related to automation of systems ([9]). From the educational point of view, Robotics is a field where many areas converge: electronics, physical (Figure 2 left), mechanical (Figure 2 right), computer sciences, telecommunications, mathematics, etc.

That is why it is a fact that Robotics is growing in importance in pre-university education, either as a field of knowledge in itself, or as a tool to present technology and other subjects to young students in an attractive way. Furthermore, Robotics has the power to motivate students and this allows us to bring technology closer to boys and girls ([10]) using robotics as a tool to present basic concepts of science ([11]), technology, engineering and mathematics (STEM) ([12]). Students learn, almost through playing, notions which are difficult and complex to explain or to assimilate through the classic masterclass ([13,14]).

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 17 October 2018 doi:10.20944/preprints201810.0372.v1

Peer-reviewed version available at *Electronics* **2018**, *7*, 430; doi:10.3390/electronics7120430

To support this increasing presence of educational robotics, there are many teaching frameworks used to teach robotics to children, from those focused on primary education to more powerful ones oriented to secondary education and high school. They are usually composed of a concrete *robotic platform*, that is to say a robot, which is programmed in a certain *language* using *software tools*. Different exercises, challenges or projects are then proposed to the students (*practice activities*). They teach the basic operation of sensors, actuators and the rudiments of programming.

2. Educational robots

The most of robots we can find among the commercial educational platforms are closed. It is worth mentioning the well known *Lego*, which has been presented for some years in educational Robotics kits, with different versions: Mindstorms RCX, NXT, EV3 and WeDo ([14,15]).

Nevertheless, *Arduino* boards appeared some years ago, in an effort to work around the closed-platforms limitation, providing cheaper and more adapted robotic platforms. This is a free hardware board which lets add a wide variety of low-cost robotic components ([16], [17], [15], [18], [19]). Thus, beginning with a basic and affordable Arduino platform, teachers and students can freely adapt it to their necessities, developing an effective and low-cost robot as described in ([20], [21], [22], [23]).

Figure 3. Robots Thymio, VEX IQ and VEX CORTEX

Another platforms are Thymio (Figure 3 left) ([24], [25], [26]), Meet Edison's or VEX robots (Figures 3 middle and right), and simulated environments such as TRIK-Studio ([19], [27]) or Robot Virtual Worlds (RVW) ([28]).

In addition, we can find different software environments. *Lego* has its own option, *EV3-software*, as Arduino does with Arduino-IDE simple text language; not to mention *Scratch* ([23], [29]) or variants: Blockly ([30]), Bitbloq or VPL. All of them contain graphic blocks that typically connect in sequence in a graphic editor. Languages such as the mentioned Arduino-IDE, or C++ (which Arduino is based on) are not suitable for pre-university students due to their complexity, but they are widely used at university level.

Exploring the existing literature we found many other works which have presented robotic platforms for educational purposes and the underlying philosophy. In [31], authors focused on a 6 Degree of Freedom (DOF) serial robotic arm as a robotic platform for training purposes. They derived the kinematic and dynamic models of the robot to facilitate the controller design. In includes an on-board camera to scan the arm workspace.

Alers and Hu showed in [32] the *AdMoVeo* robotic platform, which was developed for the purpose of teaching the industrial design students basic skills of programming. It is a platform which lets students to explore their creativity with their passions in graphical and behavioral design.

Jamieson asked in [17] whether Arduino was a platform suitable for teaching computer engineers and computer scientists an embedded system course with. He described a project based learning embedded system course that they have taught and identify which topics were covered in it compared to the *IEEE/ACM recommendations*. He finally concludes by saying that students expressed high praise for the Arduino platform and that students' final projects compared to the previous years were better and more creative.

Peer-reviewed version available at *Electronics* **2018**, *7*, 430; doi:10.3390/electronics7120430

In [33] authors presented *eBug* as a low-cost and open robotics platform designed for undergraduate teaching and academic research in areas such as multimedia smart sensor networks, distributed control, mobile wireless communication algorithms and swarm robotics. This prototype used the *Atmel AVR XMEGA 8/16-bit* micro-controller.

Miniskybot was presented in [34] as a mobile robot aimed for educational purposes which included 3D-printable on low cost reprap-like machines, fully open source (including mechanics and electronics), and designed exclusively with open source tools. It is based on an *8-bit pic16f876a* micro-controller.

Nevertheless, there is no system, and even less a guided one, that maintains a constant level of motivation and challenge, especially where vision plays an important role. In fact, the majority of these kits or robotic platforms existing in the market are focused on doing some tasks or are designed to arouse the interest of the youngest and university students in Robotics, but not so that students in pre-university courses acquire correct and complete training in programming, something which is in great demand and so widespread in almost any degree. Although it is true that other kits exist which are more specialized in specific scientific fields ([35]), the proposed framework goes further and provides all the necessary open tools for both students and teachers ([36]) required to develop a complete academic year in a versatile way by putting at their disposal numerous and sophisticated algorithms, including vision, with a pleasant and intuitive interface.

In addition, an enormous gap has been identified between the level of the academic training at university level in scientific and technological degrees and the official curriculum implemented at pre-university levels, specifically in science subjects at Secondary Education level. Thus, this work proposes to mitigate this academic gap, developing a complete teaching framework for Robotics with vision, which today is non-existent, integrating:

1. A *RaspberryPi-based open hardware platform*, economically suitable for secondary education centers to satisfy the needs of a complete class, but at the same time standardized and powerful, which allows the execution of algorithms of Robotics with vision.
2. An *open software infrastructure* that is simple and intuitive for young students to manage but that at the same time is powerful and versatile, incorporating enough resource libraries to provide practical exercises that are sufficient in both, number and complexity, on programming robots with vision, so as to continuously motivate students ([37]), as well as diverse examples.
3. A *wide repertoire of practice activities* that can be followed during a complete academic year and that includes sufficient and properly staggered sessions for correct assimilation by the students ([38]).

3. Design of the PiBot tool for STEM education

After the analysis of most relevant available educational robots, the design of the new proposed robot is described in this section. It takes benefit of some new possibilities offered by different technologies and aims to overcome some observed limitations in current platforms (like having no cameras or being not usable with programming languages like Python). It is not intended for primary education or first year secondary education, where visual languages like Scratch are better starting point. Instead it is designed for secondary education above 12 years and even introductory university courses.

Better tools improve the learning processes in kids. The `PiBot` education tool follows an architecture of three parts, as shown in Figure 4: the robot platform, the software drivers and the exercises. The robot and the drivers can be seen as the infrastructure for the exercises, which can be organized in courses or levels and focus on different aspects of robotics.

The creation of the `PiBot` tool has followed several design principles:

1. *Low cost* (under 180 euros), to make it affordable for most schools and students.
2. *Open*: first, the robot hardware should be easily assembled by the students themselves, which may also make most pieces with a 3D printer. This way the assembly of a `PiBot` can be an

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 17 October 2018 doi:10.20944/preprints201810.0372.v1

Peer-reviewed version available at Electronics 2018, 7, 430; doi:10.3390/electronics7120430

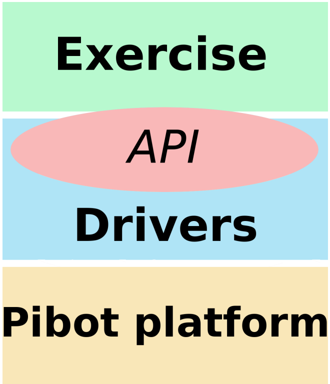

Figure 4. Architecture of the PiBot tool: hardware (platform) and software (drivers and exercise)

educative activity and interesting for the makers community. Second, drivers should be open source, publicly available.

3. Compatibility with common sensors and actuators in (arduino-based) educational robots. This way, if an Arduino-based robot is already available, the transition to `PiBot` is quite affordable; and, in any case, the acquisition of components for `PiBot` is very simple, given the large availability of components for Arduino.
4. Include *vision* in an easy way. Cameras are very useful sensors and this platform may expose students to vision in an easy and practical way.
5. It has to support not only the real robot but also a *simulated robot*. This way even with no physical platform, the `PiBot` tool may be used to teach and learn robotics.
6. *Python* as a programming language because of its simplicity, its expressive power and because it is widely used in higher levels of education and programming.

4. PiBot robotic platform

The robots are tipically composed of a computer or a microprocessor, several sensors, actuators and some form of connectivity. Sensors provide information about the environment, the computer run the robot software and actuators allow the robot to do things like moving itself or perform actions in the world.

4.1. Hardware design

The block diagram of the `PiBot` hardware is shown on Figure 5. The main computer is a Raspberry Pi 3 controller board (Figure 7 middle). It is more powerful than Arduino processors, keeps low cost, a runs a functional operating system based on Linux; specifically, the Raspbian Stretch distribution. It allows the use of standard development tools on the Linux community and the use of the PiCamera.

The sensors mounted onboard `PiBot` are:

- An ultrasound sensor model HC-SR04 (Figure 6 left)
- Infrared sensors
- Motor encoders
- Raspberry PiCamera (Figure 6 right). It is connected to the computer using a dedicated data bus. Its technical details are included in Table 1.

The US, IR and encoders sensors are connected to the RaspBerryPI board through several `GPIO` ports (*General Purpose Input/Output*). This protocol allows the connection and control of several devices at the same time and requires some configuration on each port to serve as input and output of data ([39]).

The actuators mounted onboard `PiBot` are two DC motors (Parallax Feedback 360° High Speed Servo (Figure 7, left)). They allow movement and differential drive to the PiBot. The motors include encoders and are connected to the main processor through GPIO bus.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 17 October 2018 doi:10.20944/preprints201810.0372.v1

Peer-reviewed version available at *Electronics* **2018**, *7*, 430; doi:10.3390/electronics7120430

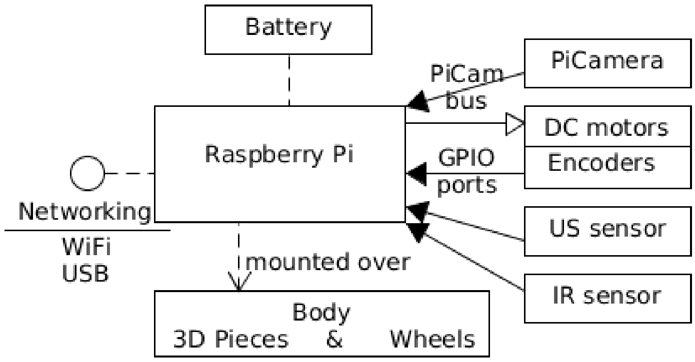

Figure 5. Hardware design of the PiBot robot

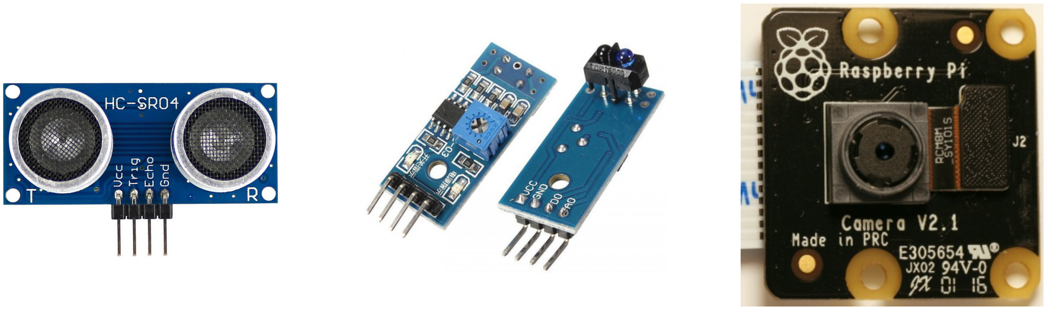

Figure 6. Ultrasonic sensor model HC-SR04, IR sensors and RaspBerry PiCamera

PiCamera params.	Values
Sensor type	Sony CMOS 8-Mpx
Sensor size	3.6x2.7mm (1/4" format)
Pixel Count	3280x2464 (active px.)
Pixel Size	1.12 x 1.12 um
Lens	f=3.04 mm, f/2.0
Angle of View	62.2x48.8 degrees
SLR lens equivalent	29 mm

Table 1. PiCamera (v2.1 board) technical intrinsic parameters

All these components are assembled into a body made of 3D printable pieces. The 3D printable models of all the chassis pieces are publicly available[1]. The body also allocates a battery of 10,000 mAh which provides power to all electronic onboard devices. An official list of components and some tentative providers are also available at the same webpage so that anyone can buy the components, print the pieces and build a PiBot.

4.2. Simulated robot

For simulation of the `PiBot` platform the Gazebo simulator [2] has been selected. It is an open source robotic simulator powered by Open Robotics Foundation and the de facto standard in the robotics scientific community. It provides a physics engine so collisions and realistic movements are provided.

[1] https://github.com/JdeRobot/JdeRobot/tree/master/assets/PiBot
[2] http://gazebosim.org

 doi:10.20944/preprints201810.0372.v1

Peer-reviewed version available at Electronics 2018, 7, 430; doi:10.3390/electronics7120430

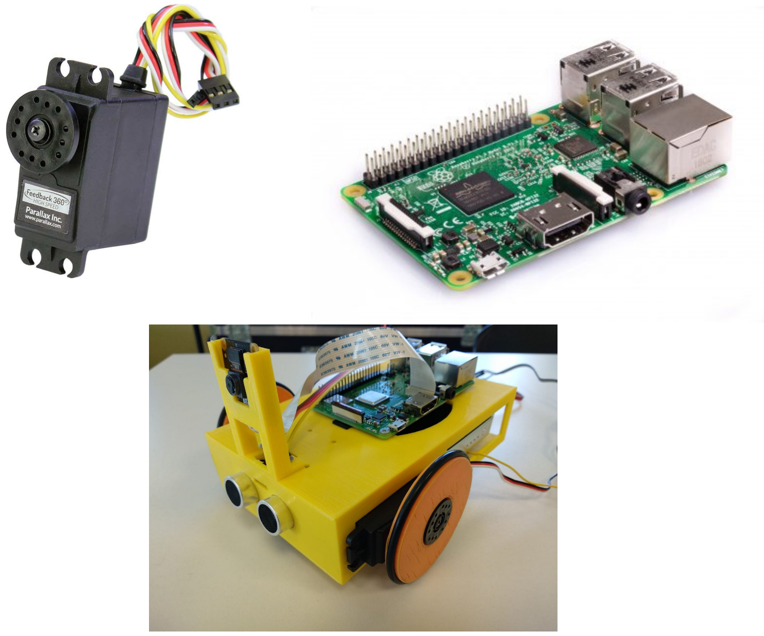

Figure 7. Motors, RaspBerryPi board and PiBot made with 3D printable pieces

The students may program an exercise and run their code seemlessly both on the physical `PiBot` or on the simulated `PiBot` inside Gazebo, at will. The student code lie on top of the `PiBot` API (Application Programming Interface), which is used to get sensor readings and to command actuator orders. The API is exactly the same on both cases. In the first one some drivers will be used to connect to the physical devices. In the second one other drivers will exchange messages with the simulator to implement the same functions.

In order to support this new robot a 3D model of the robot was developed (Figure 8). In addition, several plugins were also integrated for the simulation of the onboard camera, the distance sensor (sonar) and IR sensors. IR support has been implemented using small cameras. Each IR consists of a 4x4 pixel camera and an additional code that computes the virtual IR measurement from the values of those pixels. The movement was also supported with the corresponding Gazebo plugin, which also provides a 2D position sensor (like encoders).

The 3D PiBot model and all the developed plugins are publicly available on [3] and [4] respectively.

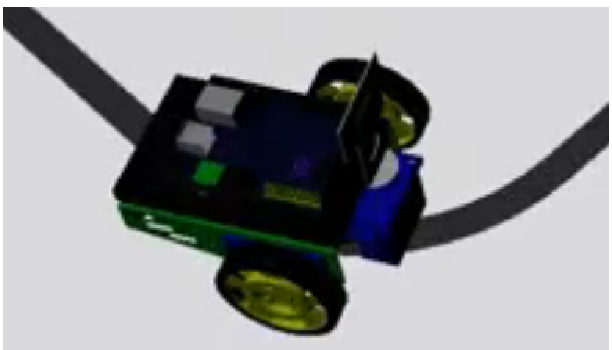

Figure 8. PiBot robot simulated in Gazebo

[3] https://github.com/JdeRobot/JdeRobot/tree/master/assets/gazebo
[4] https://github.com/JdeRobot/JdeRobot/tree/master/src/drivers/gazebo/plugins/pibot

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 17 October 2018 doi:10.20944/preprints201810.0372.v1

Peer-reviewed version available at Electronics 2018, 7, 430; doi:10.3390/electronics7120430

5. Software infrastructure

Python was chosen as a programming language for supporting `PiBot` because of its simplicity, its expressive power and because it is widely used in higher levels of education and many industries (in conjuntion with powerful libraries). It is a text language, interpreted and object oriented. This language is easier to learn than other also widely used programming languages, such as C/C++ or Java, and at the same time it has great power. It is a *real world* language but accessible for pre-university students.

With the proposed educational tool the students program their exercises in Python by writing the file `exercise.py`, for example, with a text editor. That program uses the `PiBot` Application Programming Interface (API) to control the robot, which contains a set of natural methods to read the measurements from the robot sensors (US, IR, camera) and methods to give commands to the robot actuators (DC motors). The most important API methods are detailed in Table 2.

Two different libraries have been developed to support that API. One runs onboard the `PiBot` RaspBerryPi and a second one communicates with the simulated robot inside Gazebo. As the programming interface is the same in both cases, the student application works interchangeably on the physical platform and on the simulated one. The final robot in each case is selected by specifying it on the configuration file.

Using this API, students concentrate on the algorithm they are developing, on the robot's intelligence, avoiding the low level details such as ports, connectivity with the robot, etc. which are stored in the library configuration file.

Actuators	Sensors
RightMotor(V) LeftMotor(V)	readUltrasound readInfrared getImage
move(V, W)	getColoredObject(color) getDistancesFromVision getRobotPosition

Table 2. Application Programming Interface (API)

The API methods can be divided into raw methods and cooked methods. Raw methods provide access to a single device, like readUltrasound, readInfrared or getImage. RightMotor(V) controls the single right motor commands a desired speed to it, as LeftMotor(V) does for the other motor. The cooked methods provide a simpler and more compact way to control the whole robot or two vision functions to get useful information from the image in an easy way. They will be detailed later.

5.1. Drivers for the real PiBot

To support the real `PiBot` two modules were programmed as shown in Figure 9. One includes the management of the PiCamera and the other deals with GPIO devices (US sensor, IR sensors and motors). They were programmed in Python using standard available libraries in Python community. It is publicly available[5]. The image processing functionality also relies on OpenCV.

5.2. Drivers for the simulated PiBot

To support the simulated PiBot on Gazebo an specific library was developed which connects with the plugins mentioned before exchanging messages through the ICE communication layer. It achieves

[5] https://github.com/JdeRobot/JdeRobot/tree/master/src/drivers/PiBot/real

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 17 October 2018 doi:10.20944/preprints201810.0372.v1

Peer-reviewed version available at Electronics 2018, 7, 430; doi:10.3390/electronics7120430

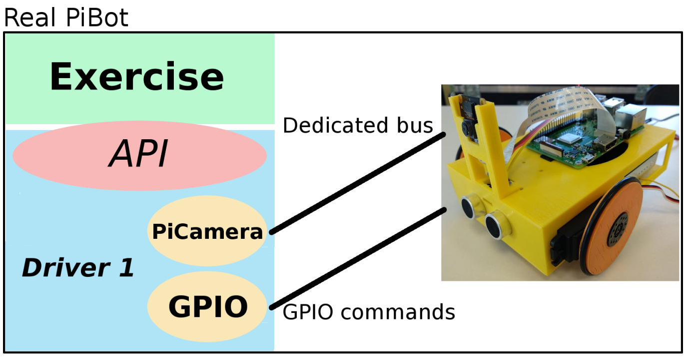

Figure 9. Connection of the library with the real PiBot

sensor readings and camera images through network interfaces built in the JdeRobot project [6]. It is also publicly available [7].

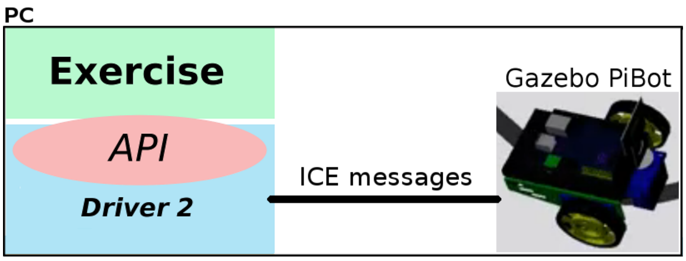

Figure 10. Connection of the library with the simulated PiBot

5.3. Movement control

Regarding motors, beyond the raw methods `RightMotor(V)` and `LeftMotor(V)` a new cooked method is provided for simpler control of the robot movements: `Move(V,W)`. This method accepts as parameter the desired lineal speed V and the desired rotation speed W, it internally translates them into commands to the left and right motors so that the whole robot moves accordingly to V and W. It takes into account the geometry of the PiBot and its wheels.

This function provides general 2D movement control: the PiBot may rotate without displacement (setting $V = 0$ and using W) both left or right (depending on the sign of W), may advance in straight line (setting $W = 0$ and using V) both backwards and forward (depending on the sign of V), and may move in generic arcs advancing and rotating at the same time.

It is a speed control which is useful when programming reactive behaviors, which is better that position-based control.

5.4. Vision support

One advantage of PiBot educational tool is its support for the camera. This allows many new possible exercises with vision and vision-based behaviors. It also introduces the students to computer vision in a simple and natural way. Two functions (`getColoredObject(color)` and `getDistancesFromVision`) have been included so far in the PiBot API to get easily useful information from images, because the raw method `getImage` and the pixels processing are too complex for high school students. They have been implemented and included in a vision library which performs complex image processings, hides all the complexity inside and it is really simple to use, very intuitive. It internally employs OpenCV library, a standard in Computer Vision community.

[6] https://jderobot.org
[7] https://github.com/JdeRobot/JdeRobot/tree/master/src/drivers/PiBot/Gazebo

Peer-reviewed version available at Electronics 2018, 7, 430; doi:10.3390/electronics7120430

First, the cooked method `getColoredObject(color)` accepts the desired *color* as input parameter and it filters in the current camera image all the pixels of that color (some of them are already predefined in the library: orange, red, blue...). It delivers as output the position of the colored object inside the image (its mean X and Y value) and its size (the number of detected pixels of that color). It works with single objects as can be seen in Figure 11.

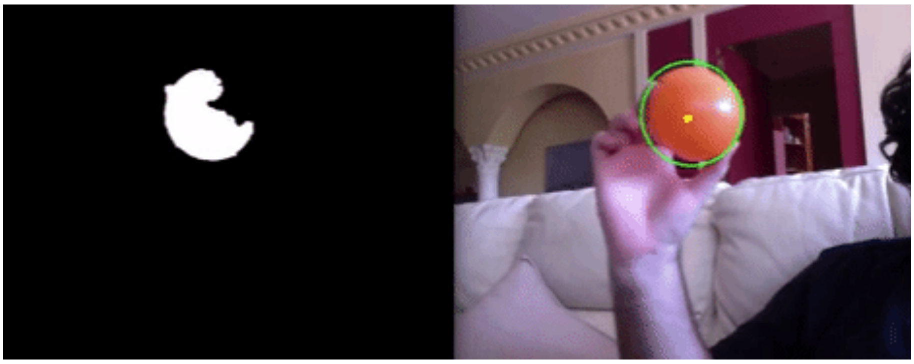

Figure 11. GetColoredObject function for orange color

It uses HSV color space and OpenCV filtering methods. This function on PiBot API allows for exercises like Object-Following, which will be detailed in the next section.

Second, the cooked method `getDistancesFromVision` computes the distance to obstacles in front of the PiBot and provides a depth map from the robot to the surrounding objects. Typically the sonar sensor measures the distances in one direction. Using the camera for the same the angular scope is extended to the camera field of view (around 60 degrees).

The developed vision library contains an abstract model of the camera (pin-hole) and several projective geometry algorithms. The camera parameters are known (K matrix and relative position inside the robot). As the PiBot only has a single camera no stereo technique can be used for depth estimation. Instead, the implementation of `getDistancesFromVision` method assumes that all objects lie on the floor and the floor surface has a uniform color (*ground hypothesis*). It sweeps all the columns of the current image from its bottom. When the first edge pixel is found on a column it is backprojected into 3D space, using ray tracing and the pin-hole camera model. The intersection of such ray with the floor plane is the estimated position of that edge in 3D space, and its distance to the robot is computed. In this way, the 3D point corresponding to each bottom pixel of the obstacle in the image can be obtained (Figure 12).

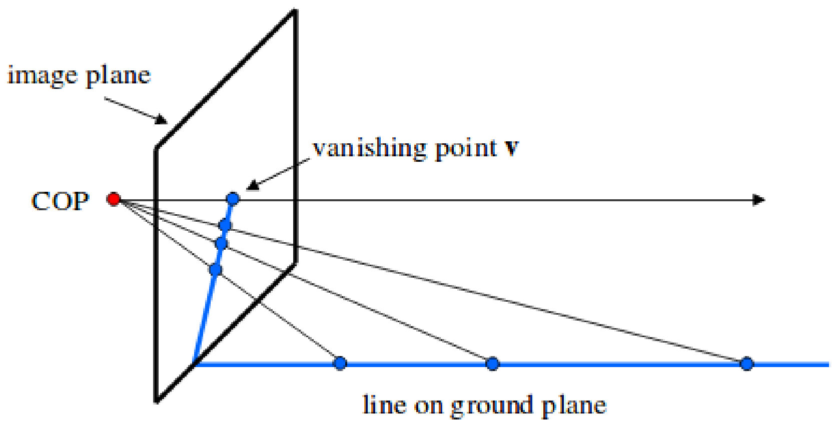

Figure 12. Ground Hypothesis assumes all objects are on the floor

For instance, Figure 13 shows in the left side the image coming from the camera, with the white floor (the appearing battery was be safely ignored as only green pixels were taken into account for

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 17 October 2018 doi:10.20944/preprints201810.0372.v1

Peer-reviewed version available at *Electronics* **2018**, *7*, 430; doi:10.3390/electronics7120430

explanatory purposes in this test). On the right side the estimated depths for the green object are displayed as red points and the field of view is also shown as a white trapezoid. The estimated distances are regularly consistent and correct.

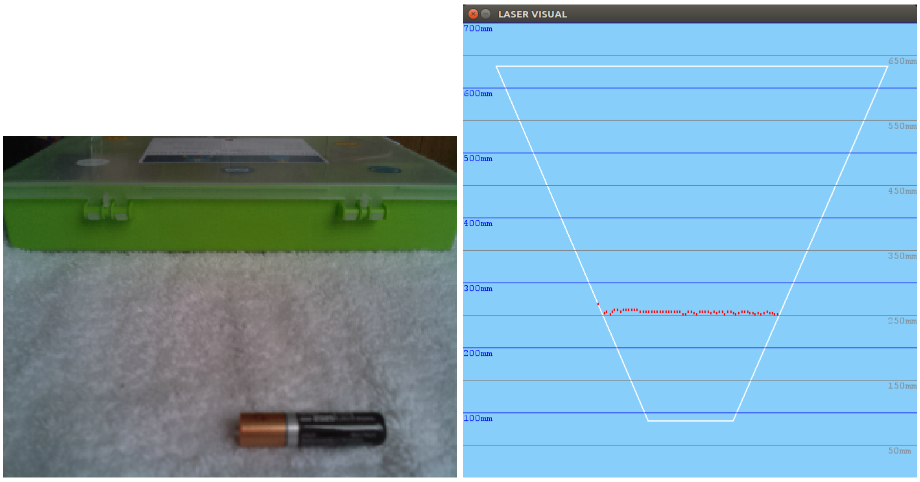

Figure 13. Example of visual sonar reading with 25 cm object shown using 3D scene simulator

This `getDistancesFromVision` function on PiBot API allows for exercises like robot navigation with obstacles. For instance the vision-based obstacle avoidance which will be detailed in the next section.

6. Exercises for students using PiBot

Finally, a plan of activities using the `PiBot` is described.

6.1. Basic exercises

Students can begin assembling different components on the `PiBot` and review some basic concepts of electronics so that they have no problems when connecting the different components, such as infrared or ultrasound sensors (Figure 14).

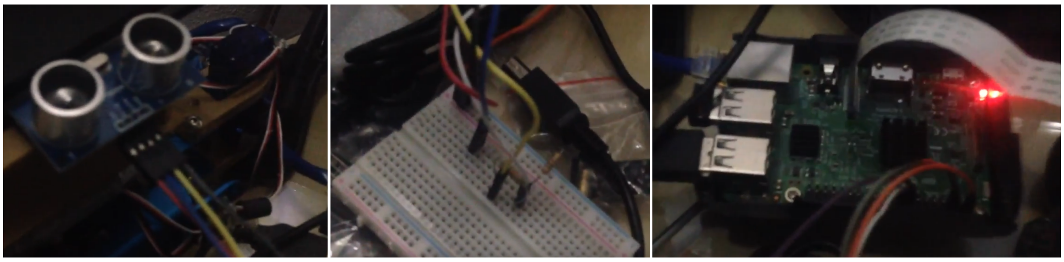

Figure 14. Practice activity with PiBot to handle an ultrasonic sensor

6.2. Basic behaviors

The last step consists of a complete Robotics project where students can combine everything previously learnt. Firstly, they can begin developing classic Robotics projects like line tracking using infrared sensor (Figure 15) or navigation avoiding obstacles by means of ultrasounds (Figure 16).

6.3. Vision-based behaviors

Secondly, students can develop more advanced robotics projects, using vision as the main sensor. Some projects developed are: following an colored object (Figure 17), line tracking (Figure 18) or bump-and-go (Figure 19).

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 17 October 2018 doi:10.20944/preprints201810.0372.v1

Peer-reviewed version available at Electronics 2018, 7, 430; doi:10.3390/electronics7120430

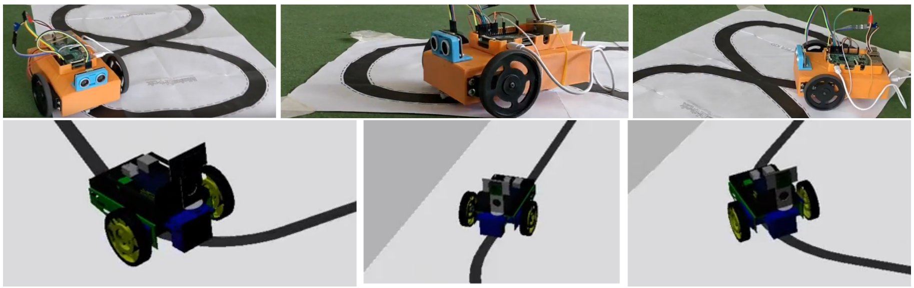

Figure 15. Practice line tracking task in both real and simulated PiBot platforms using IR sensor

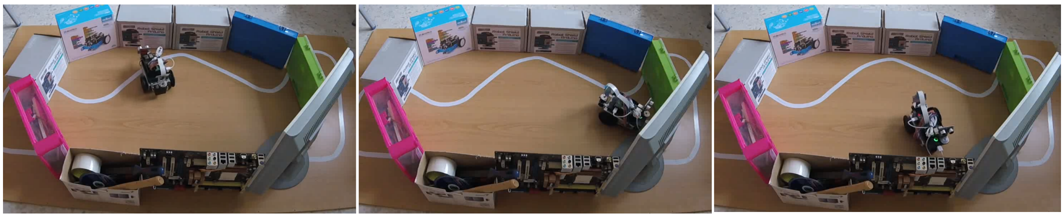

Figure 16. Navigation practice avoiding obstacles through US in PiBot

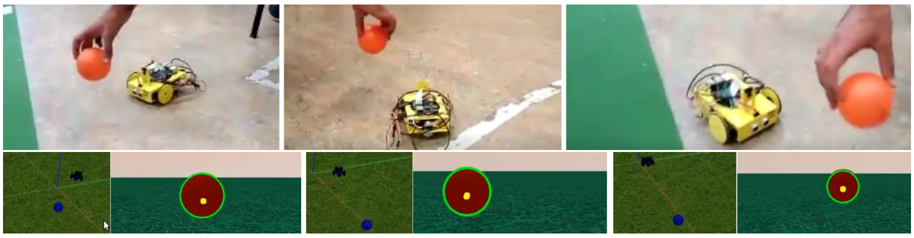

Figure 17. Navigation exercise of following a colored object using vision in both real and simulated PiBot platforms

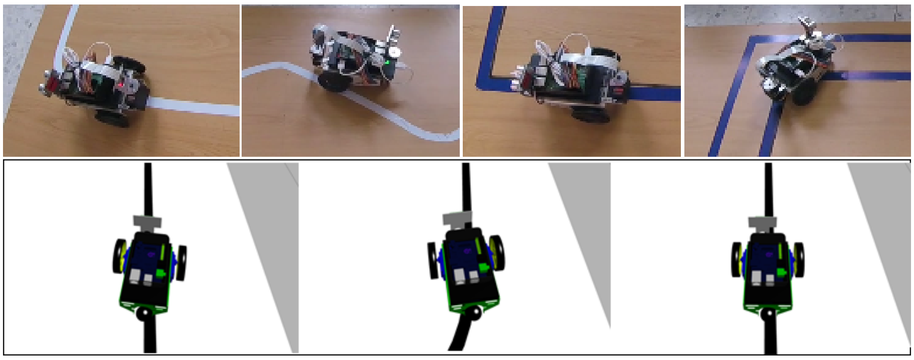

Figure 18. Practice line tracking task in both real and simulated PiBot platforms using vision

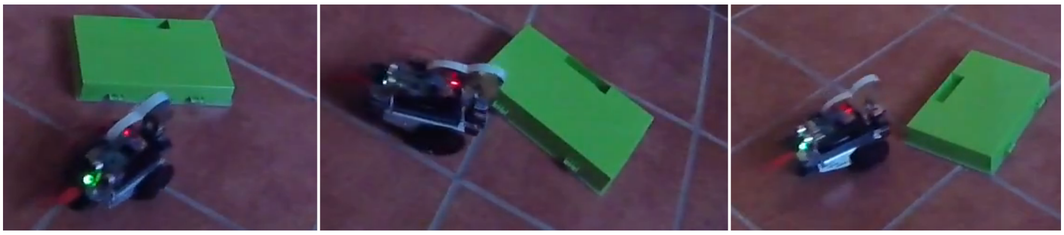

Figure 19. Exercise of avoiding obstacles by visual depth estimation in PiBot

305 During the last month of July a Robotics workshop was taught to ten teachers at the Campus
306 of Fuenlabrada of the Rey Juan Carlos University (Madrid) (Figure 20), training them to use the
307 developed framework with `PiBot` as a robotic platform.

Peer-reviewed version available at *Electronics* **2018**, *7*, 430; doi:10.3390/electronics7120430

Figure 20. Workshop at Rey Juan Carlos University to train teachers for teaching with JdeRobot-Kids framework using PiBot

7. Conclusions

This research is focused on incorporating Robotics and robots with vision in the classroom to train pre-university students, satisfying the demands imposed by the *Digital Age Society* and the motivational needs detected in students, who still study in a system of training still to be adapted to the so-called *Industrial Revolution 4.0*.

Although there are numerous educational Robotics kits on the market, most of them are aimed at very young students. They are generally based on building *their* robotic platforms with *their* own programming environments, far from employing more standardized programming languages. They usually have somewhat limited capabilities which means that these tools tend to trigger a low level of motivation in students in the mid term (for instance in students that have already followed an introductory robotics course). Furthermore, given the complexity involved in the processing of images, cameras are not usually included in the educational robotic frameworks despite their great versatility and extensive use in real life applications.

After studying the current market of the existing Robotics educational kits and conducting an in-depth analysis what the future holds in the short and mid-term in terms of demands of the labor market, the authors (one of them an experienced Secondary Education teacher) detected a deficiency in the teaching-learning process of Robotics at pre-university curricular level. Therefore, a complete new educational tool was developed, which includes:

- A robotic platform based on the free hardware controller board Raspberry Pi 3. This platform was chosen for several reasons: low cost, power, versatility, standardization and inclusion of a camera with its own data bus, the `PiCamera`. Thus, a fully functional robot, the `PiBot`, and the counterpart for Gazebo simulator and for *DIY* 3D printable model were developed. Thanks to the GPIO ports on the board, various sensors and actuators —both real and simulated— have been connected, in addition to its own camera.
- A software infrastructure developed in Python language, which facilitated students' programming of the robot, with simple and intuitive functions to handle the different sensors and actuators. At the same time this infrastructure has great potential corresponding to its handling of a camera as a sensor.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 17 October 2018 doi:10.20944/preprints201810.0372.v1

Peer-reviewed version available at *Electronics* **2018**, *7*, 430; doi:10.3390/electronics7120430

- A wide set of exercises that serve as a support to students for their progression in the learning of the programming of robots with vision.

About future lines, one intended improvement in the short term are is to extend the vision support: (a) developing new practical sessions with vision such as the detection and monitoring of people's faces, and materialize in the `PiBot` a visual memory; (b) the camera may also be seated on a servo and so the current vision range could be extended to a wider field of view, thanks to the movement of the camera.

It is also intended to develop the support for the encoders of the `PiBot` motors, which would allow to develop more position-based sophisticated navigation.

Finally, authors are also working to support `PiBot` programming with the popular visual Scratch language, so that younger students can start programming this robot in a very simple way. With the same `PiBot` platform they could start learning robotics with Scratch and later on jumpo to Python and face more appealing exercises.

1. Mies, G.; Zentay, P. Industrial Robots meet industry 4.0. XII Hadmernok, 2017.
2. Schwab, K. *The Fourth Industrial Revolution*; World Economic Forum, 2016.
3. Vega, J.; Cañas, J. Sistema de atención visual para la interacción persona-robot. Workshop on Interacción persona-robot, Robocity 2030, pp. 91-110. ISBN: 978-84-692-5987-0, 2009.
4. Arbel, T.; Ferrie, F. Entropy-based gaze planning. *Image and Vision Computing, vol. 19, no. 11, pp. 779-786* **2001**.
5. Solove, D. *The Digital Person: Technology and Privacy in the Information Age*; 2004.
6. Frey, C.; Osborne, M. *The future of employment: how susceptible are jobs to computerisation?*; 2013.
7. Deloitte. *From brawn to brains: The impact of technology on jobs in the UK*; 2015.
8. Institute, M. Jobs lost, jobs gained: workforce transitions in a time of automation, 2017.
9. UK-RAS. Manufacturing Robotics: The Next Robotic Industrial Revolution, 2016.
10. Rodger, S.H.; Walker, E.L. Activities to attract high school girls to computer science. *National Science Foundation's Directorate for Education and Human Resources under the Model Projects for Woman and Girls* **1996**.
11. Altin, H.; Pedaste, M. Learning approaches to applying robotics in science education. *Journal of baltic science education* **2013**, *12*, 365–377.
12. Mubin, O.; Stevens, C.J.; Shahid, S. A review of the applicability of robots in Education. *Technology for Education and Learning, 2013* **2013**.
13. Cerezo, F.; Sastrón, F. Laboratorios Virtuales y Docencia de la Automática en la Formación Tecnológica de Base de Alumnos Preuniversitarios. *Revista Iberoamericana de Automática e Informática Industrial RIAI* **2015**, *12(4)*, 419–431. doi:10.1016/j.riai.2015.04.005.
14. Jiménez, E.; Bravo, E.; Bacca, E. Tool for experimenting with concepts of mobile robotics as applied to children education. *IEEE Trans. Education* **2010**, *53*, 88–95.
15. Navarrete, P.; Nettle, C.J.; Oliva, C.; Solis, M.A. Fostering Science and Technology Interest in Chilean Children with Educational Robot Kits. Robotics Symposium and IV Brazilian Robotics Symposium (LARS/SBR), 2016 XIII Latin American. IEEE, 2016, pp. 121–126.
16. Araujo, A.; Portugal, D.; Couceiro, M.S.; Rocha, R.P. Integrating Arduino-Based Educational Mobile Robots in ROS. *J Intell Robot Syst (2015) 77:281-298* **2015**.
17. Jamieson, P. Arduino for Teaching Embedded Systems. Are Computer Scientists and Engineering Educators Missing the Boat? *Miami University, Oxford, OH, 45056* **2012**.
18. Chaudhary, V.; Agrawal, V.; Sureka, P.; Sureka, A. An experience report on teaching programming and computational thinking to elementary level children using lego robotics education kit. Technology for Education (T4E), 2016 IEEE Eighth International Conference on. IEEE, 2016, pp. 38–41.
19. Filippov, S.; Ten, N.; Shirokolobov, I.; Fradkov, A. Teaching robotics in secondary school. *IFAC-PapersOnLine* **2017**, *50*, 12155–12160.

Peer-reviewed version available at *Electronics* **2018**, *7*, 430; doi:10.3390/electronics7120430

20. Junior, L.A.; Neto, O.T.; Hernandez, M.F.; Martins, P.S.; Roger, L.L.; Guerra, F.A. A low-cost and simple arduino-based educational robotics kit. *Cyber Journals: Multidisciplinary Journals in Science and Technology, Journal of Selected Areas in Robotics and Control (JSRC), December edition* **2013**, *3*, 1–7.
21. Plaza, P.; Sancristobal, E.; Fernandez, G.; Castro, M.; Pérez, C. Collaborative robotic educational tool based on programmable logic and Arduino. Technologies Applied to Electronics Teaching (TAEE), 2016. IEEE, 2016, pp. 1–8.
22. Afari, E.; Khine, M. Robotics as an educational tool: Impact of LEGO mindstorms. *IJIET* **2017**, *7*, 437–442.
23. Beyers, R.N.; van der Merwe, L. Initiating a pipeline for the computer industry: Using Scratch and LEGO robotics. Information Communication Technology and Society (ICTAS), Conference on. IEEE, 2017, pp. 1–7.
24. Mondada, F.; Bonani, M.; Riedo, F.; Briod, M.; Pereyre, L.; Rétornaz, P.; Magnenat, S. Bringing robotics to formal education: the thymio open-source hardware robot. *IEEE Robotics & Automation Magazine* **2017**, *24*, 77–85.
25. Roy, D.; Gerber, G.; Magnenat, S.; Riedo, F.; Chevalier, M.; Oudeyer, P.Y.; Mondada, F. IniRobot: a pedagogical kit to initiate children to concepts of robotics and computer science. RIE 2015, 2015.
26. Magnenat, S.; Shin, J.; Riedo, F.; Siegwart, R.; Ben-Ari, M. Teaching a core CS concept through robotics. Proceedings of the 2014 conference on Innovation & technology in computer science education. ACM, 2014, pp. 315–320.
27. Stone, A.; Farkhatdinov, I. Robotics Education for Children at Secondary School Level and Above. Conference Towards Autonomous Robotic Systems. Springer, 2017, pp. 576–585.
28. Witherspoon, E.B.; Higashi, R.M.; Schunn, C.D.; Baehr, E.C.; Shoop, R. Developing computational thinking through a virtual robotics programming curriculum. *ACM Transactions on Computing Education (TOCE)* **2017**, *18*, 4.
29. Plaza, P.; Sancristobal, E.; Carro, G.; Castro, M.; Blázquez, M.; Muñoz, J.; Álvarez, M. Scratch as Educational Tool to Introduce Robotics. International Conference on Interactive Collaborative Learning. Springer, 2017, pp. 3–14.
30. Naya, M.; Varela, G.; Llamas, L.; Bautista, M.; Becerra, J.C.; Bellas, F.; Prieto, A.; Deibe, A.; Duro, R.J. A versatile robotic platform for educational interaction. Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS), 2017 9th IEEE International Conference on. IEEE, 2017, Vol. 1, pp. 138–144.
31. Manzoor, S.; Islam, R.U.; Khalid, A.; Samad, A.; Iqbal, J. An open-source multi-DOF articulated robotic educational platform for autonomous object manipulation. *Robotics and Computer-Integrated Manufacturing* **2014**, *30*, 351 – 362.
32. Alers, S.; Hu, J., AdMoVeo: A Robotic Platform for Teaching Creative Programming to Designers. In *Learning by Playing. Game-based Education System Design and Development*; 2009; pp. 410–421.
33. Dademo, N.; Lik, W.; Ho, W.; Drummond, T. eBug - An Open Robotics Platform for Teaching and Research. *Proceedings of Australasian Conference on Robotics and Automation* **2011**.
34. Gonzalez, J.; Valero, A.; Prieto, A.; Abderrahim, M. A New Open Source 3D-printable Mobile Robotic Platform for Education. *Advances in Autonomous Mini Robots* **2012**.
35. Schweikardt, E.; Gross, M.D. roBlocks: A Robotic Construction Kit for Mathematics and Science Education. Proceedings of the 8th International Conference on Multimodal Interfaces, 2006, ICMI '06.
36. Bers, M.U.; Ponte, I.; Juelich, C.; Viera, A.; Schenker, J. Teachers as Designers: Integrating Robotics in Early Childhood Education. *Information Technology in Childhood Education Annual* **2002**, *2002*, 123–145.
37. Benitti, F. Exploring the educational potential of robotics in schools: A systematic review. *Computers and Education* **2012**, *58*, 978 – 988.
38. Ainley, J.; Enger, L.; Searle, D., Students in a Digital Age: Implications of ICT for Teaching and Learning. In *International Handbook of Information Technology in Primary and Secondary Education*; Voogt, J.; Knezek, G., Eds.; Springer US: Boston, MA, 2008; pp. 63–80.
39. Balachandran, S. General Purpose Input Output (GPIO). Technical report, ECE 480 Design team 3. Available in the College of Engineering, Michigan State University website, 2009.