Remote control and autonomous vision-guided navigation system for a hexapod robot

1st Miguel Ángel Anguita-Molina
Department of Computer Science
University of Jaén
Jaén, Spain
mamolina@ujaen.es

2nd Javier Medina-Quero
Department of Computer Science
University of Jaén
Jaén, Spain
jmquero@ujaen.es

3rd Silvia Satorres Martínez
Department of Electronics and Automation
University of Jaén
Jaén, Spain
satorres@ujaen.es

4th Aurora Polo-Rodríguez
Department of Computer Science
University of Jaén
Jaén, Spain
apolo@ujaen.es

***Abstract*—Robots are increasingly present in our daily lives. This, together with the processing power available in small devices and the amount of information that can be obtained from a vision sensor, allow to create robotic systems with a great variety of functionalities. Therefore, in this work we propose the design, development and integration of: i) hardware components to build the new version of the robot, ii) communication methods between sensors and boards of Arduino and Raspberry Pi and a remote service for controlling the hexapod robot, iii) an autonomous navigation system based on marker recognition.**

***Index Terms*—vision-guided robot, computer vision, hexapod robot, autonomous navigation**

I. Introduction

The number of devices involved in machine-to-machine (M2M) communications has grown steadily in recent years, making the anticipated revolution known as the Internet of Things (IoT) a reality. One of the main proclaimed goal in IoT is "to connect everything and everyone everywhere to everything and everyone else" [1]. In addition, robots are playing a major role in today's society, continuing to help humans in accomplishing many duties. Research and application trends are leading to the appearance of the Internet of Robots [2], and to IoT-aided robotics applications. Most of modern robots are equipped with sensing, computing, and communication capabilities, which make them able to execute complex and coordinated operations [3].

Amongst other criteria, robots can be classified according to: i) the type of task that they will perform (industrial vs. service), ii) the material of which they are made of, iii) their configuration (cartesian, articulated, parallel, etc), iv) type of control system, or if they are free or closed software. However, the determining factor is the task that the robot has to perform, since the same materials or actuators determine if they are suitable as a hobby robot or as a rescue robot in a natural disaster. Currently, the use of robots, specifically hexapod robots, have attracted considerable attention because they have several benefits as stability, flexibility, efficiency, robustness and performing complex operations [4].

Focusing on performing complex operations, Land founds that when humans perform a complex motor task, the oculomotor system keeps the centre of gaze very close to the point at which information is extracted, also known as visual fixation [5]. This system should be able to operate from visual feedback, succeed without the need for camera calibration, and be flexible enough to tolerate task variations, to be truly robust much like humans [6].

The use of visual information in the control loop feedback of a robotic system is known as visual servoing. Main scenarios where vision-based robots are used in warehousing industry, logistics and transportation systems [7], but vision-based mobile robot navigation technology has been a new research boom in recent years and is one of the essential directions of mobile robot guidance technology research work [8].

To bring these fields together, in this work, we describe a remote control and autonomous visual servoing system for a hexapod robot based on the next highlights:

- We start from the hexapod robot which was endowed with different degrees of freedom and different speed levels. The robot had a built-in PS2 controller that communicated through an own designed Printed Circuit Board (PCB), which was connected to the Arduino in the robot.
- Integration of vision sensor and Raspberry Pi which allowed us to give the recognition functionality to the robot. This is a basic and a complex function for computer vision, by means of which the system is able to learn to recognize shapes in order to classify them correctly [9].
- Integration of REST services for the remote control of the hexapod robot. It is a set of definitions and protocols that were used to develop and to integrate application software, enabling communication between two software

European Union's Horizon 2020 research and innovation programme under the grant agreement No 85718

applications through a set of rules [10], which allowed us to communicate the robot with a computer.
- Deployment of a vision-guided navigation system based on fiducials marks. The visual information of the scene was provided by the ArUco marks [11]. Basically, it consist of a fiducials marker system, commonly used for camera localization and tracking when robustness, precision, and speed are required [12].

As a summary of this work, we describe the integration of different methodologies such as IoT boards, hexapod robots, vision sensors and artificial intelligence, which allowed us to develop the independence of our robot in order to be deployed in tagged smart worlds.

The structure of the paper is as follows: in Section 2, the key findings from the literature review regarding visual servoing in hexapod robots are presented. Section 3 details the hardware and software implementations developed for the autonomous navigation of the hexapod. Section 4 shows how the robot, using information acquired from the scene, is able to follow a predefined path. Finally, Section 5 presents the conclusions and future works.

II. Related work

Initially, robotic systems incorporating computer vision worked in an incremental open-loop. This technique is known as *look and move*, i.e., first the robot sees and recognizes the environment helped by a computer vision system, and after that, it performs the motion based on the data acquired in the previous step [13]. The *look-and-move* system supposes that the object position, obtained by the vision system, is not altered from the moment it was acquired until the robot reaches the reference position [14].

An alternative to the previous approach is visual servoing (or visual servo control). Visual servoing is based on the use of visual information in the control loop feedback. This closed-control loop approach permits to correct possible errors in the object position estimation obtained from the computer vision system. Moreover, it permits to change the robot trajectory in view of possible movements of the objects in the workspace [15]. A large number of robotics problems can be approached with visual servoing techniques, examples of this include manufacturing and surgery, among many others [16].

A good example that reached the whole world and aroused great curiosity for this field was the Spot robot from Boston Dynamics, this quadruped is being used from exploration in extreme environments, to help fight COVID-19 [17].

In this work, we focused on visual servoing for navigation. In the broader context of navigation task, visual servoing, discussed here, can be viewed as local navigation where the goal is to reach the desired view or navigate towards the target of interest [18]. A proposal is the Perseverance rover, which is carrying out a mission to obtain samples on Mars. The robot needs visual servoing from its entry on Mars, since, in order to land, it needed to locate its position by means of cameras and land in the designated place. The rover has 23 cameras on board, some of them only serve to show images of Mars [19], but the others are used for visual servoing.

In our case, we used the previously mentioned ArUco marks to set those targets that have to be reached. Babinec reported a localisation method for mobile robots with the use of the ArUco markers deployed to the environment [20]. The results showed that this system was reliably employed in the visual localisation of mobile robots. Xing developed a multi-sensor fusion indoor localisation system based on the ArUco markers for mobile robotics. The sensors include: markers, optical flow, ultrasonic and the inertial sensor. The results showed that the proposed method has satisfactory performances [21]. In [22], the extended Kalman filter algorithm was utilised to fuse odometer information and the information from detection ArUco markers. Meng applied the ArUco markers to provide localisation services for indoor IoT (Internet of Things) applications [23]. Zhizun used Aruco markers to research how to make an underwater mobile robot with which he achieved great results [24].

Another example, but this time taking advantage of microgravity, is the Astrobee, which is a new free-flying robotic system to perform Intravehicular Activity (IVA) work on the International Space Station (ISS). The Astrobee system includes three free-flying robots, a dock (for recharging electrical power and transferring large data files), and a ground data system used for communication, control, and data transfer [25]. These robots use ArUco markings to locate themselves in space, which gives confidence that the technologies being used are state-of-the-art and are being worked with right now in high-tech environments.

Regarding IoT, also utilized in this work, most of the initiatives are focused on connecting devices with simple onboard passive sensors to manage, monitor and optimize systems and their processes. Even though impactful, the potential of IoT solutions could be further unlocked by exploring the more advanced and transformational aspects of ubiquitous connectivity to, and communication among, smart devices.

Robotic systems can aid this transformation because of their inherent ability to sense, think (compute), act (manipulate) and move around (mobility). Internet of Robotic Things (IoRT) [26] is a new concept given by ABI Research, that depicts this synergistic nature between IoT and robotics, where intelligent devices can monitor events, fuse sensor data from a variety of sources, use local and distributed intelligence to determine a best course of action, and then act to control or manipulate objects in the physical world, and in some cases while physically moving through that world.

IoT-aided robotics applications will grow upon a digital ecosystem where humans, robots, and IoT nodes interact on a cooperative basis. The synergy of IoT and robotics remains largely an untapped field of future technology that has the potential to bring about drastic changes to how we live today. IoT based solutions are changing the way we tackle problems. Smart homes, wearables, smart cities, smart grids, industrial internet, connected cars, connected health, smart retail, smart supply chains and smart farming are only a few of the IoT

applications in daily times which have impacted how we live as a society. By providing real time, quantifiable and decisive data, IoT has reduced our response time to critical problems and in a few cases made removed the need for human supervision to solve problems [27].

On communication protocol, API REST has been chosen due to the great number of clients that offer application programming interfaces (APIs) to connect with these services. Several studies point out that developers have moved from simple object access protocol (SOAP) or remote procedure call (RPC) to deploying representational state transfer (REST) services as a means for consumers to use their services. This is corroborated by large websites such as Google, Facebook or Twitter, which are now deploying REST services to facilitate access to their valuable data resources while promoting their businesses [28].

Based on the works and approaches reviewed in this section, this work presents the adaptation and development of an autonomous navigation system for a hexapod robot, giving it total freedom to move in a marked environment, making a synergy between a visual servoing system and IoRT.

III. Methods

A. Hardware

The objective of this work was to incorporate a computational and communication capabilities in the hexapod robot, that is a robot developed in a past work [29], called Uja-Spider-Robot (USR). In this section, we focus on describing hardware, communication and path algorithm based on visual mark navigation. The new version of the vision-based hexapod robot is called USR++.

B. Hardware proposal

We started from the configuration of the hexapod robot USR which previously integrates an Arduino board which acts as primitive spinal cord and the servomotors acting as the muscles. The robot is powered by a 6000 mAh battery, which gave a current of 5.5V and 2A, as for the control of the robot, as indicated above, it was controlled with a PS2 controller as it is shown in Figure 1.

The robot had several modes of operation, including displacement and body control. Every operation mode provided a wide range of movements and rotations, which worked to failure. Special attention had to be paid for not to exceed the limits of mobility.

The failure was caused when an attempt was made to place the legs in a position which implied a height greater than their wingspan, or in case that one of the joints needed to be oriented at a certain angle which was outside the limits.

First, we increased the vision sensor capabilities of the hexapod robot USR using a Raspberry Pi with an infrared camera, NoIR model, connected directly to the Arduino through a serial port. A script allowed the communication between these two elements. The Raspberry Pi can be considered the brain of the hexapod robot.

As mentioned above, the robot had two modes of movement, displacement and body control. In the travel mode, the hexapod moved freely on the floor and can also rotate and increase its height with respect to the floor. By contrast, in the body control mode, the rotation and translation of the robot on the three coordinate axes was controlled.

Fig. 1. Hexapod robot.

The hexapod integrates an own designed PCB as a shield connected to the Arduino (Figure 2). It was necessary to remove two connections, Vcc and GND, to power the RaspBerry with GPIO connections. These connections were removed by jumpers, which go from the 5.5V and GND connections on the right side of the PCB to pins 2 and 14 of the RaspBerry Pi.

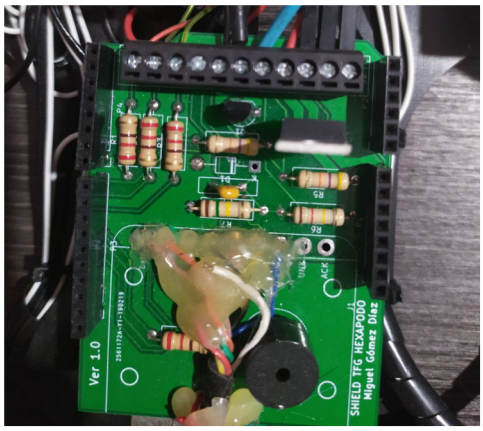

Fig. 2. Own designed Robot PCB.

Next, we connected the NoIR Camera Module V2, which are the eyes of the hexapod robot. It is a high definition infrared camera with automatic control functions, such as lighting detection. To finish the hardware part, we connected the RaspBerry via USB with the Arduino, to be communicated by serial port. From the serial port, we were able to command from the two IoT boards in order to integrate a unique schema as it is shown in Figure 3.

C. API REST for remote controlling vision sensor and servomotors

In this section, we describe the integration of API REST services which we developed for remote controlling vision

Fig. 3. Hexapod robot after incorporating new elements.

sensor and servomotors of the hexapod robot, and how these services allowed us to communicate a visual interface with USR++.

The main objective is to control through the REST services, to which clients connects URL, in order to move the hexapod robot USR++. Internally, the communication process is described in these layers:

- Python API Server in Raspberry. The program receives, by means of the URL to which the user connects, the movement that he wants to send to the Arduino, takes the corresponding part of the URL and sends it by serial port to the Arduino. Movements were mapped as WASD. W/S controls forward and reverse and A/D controls left and right.
- Raspberry Pi sends by serial port to the Arduino, where an integrated program in C++ within Arduino receives the movement commands, processes it and, depending on what it receives, the robot will perform one movement or another.
- The REST server receives the images through the camera connected to the RaspBerry, takes those images and process them, giving as a result the identifier, the position and the rotation with respect to the ArUco mark.

The processed image was also displayed via HTTP protocol, in order to visualize if the algorithm was working correctly. The algorithm indicates to the robot the mark that has to be reached. Once the robot reaches the point, it will communicate to the algorithm that it has arrived and, at that moment, the algorithm continues its execution and indicates the next mark to be reached. The procedure is recursively executed until the robot finishes its task.

Since this project was intended for educational activities to promote interest in electronics and computer science, it was decided to include a graphical interface simulating the surface of Mars in 3D (Figure 4). The simulation was developed in Unity and various elements, both decorative and functional, were added. Decorative elements included buildings, such as a dome or a Mars base, and structures such as an arch. Apart from the virtual USR++ itself, meteorites were introduced as interactive elements, symbolising the position of the real-life landmarks, to which the hexapod had to go in both worlds. There were also different viewpoints to visualise the whole scenario.

Our robot can be controlled from the 3D graphical interface in real time, thanks to API/REST services mentioned above, so that all movements made in the virtual world are reflected in the real world, including autonomous navigation mode. Unity's path resolution algorithm and simulation communicate via Message Queuing Telemetry Transport (MQTT) with both the physical USR++ and the simulation USR++. This allows simultaneous routing based on the routing algorithm detailed below.

Fig. 4. Third-person view of the simulation.

With all the above elements, in addition to the pathfinding algorithm, we obtained a communication scheme shown in Figure 5.

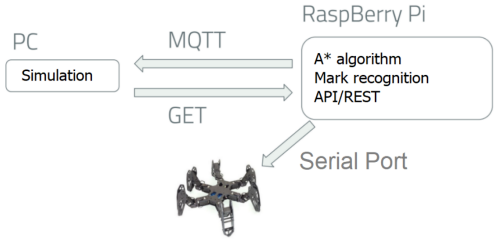

Fig. 5. Communication scheme.

D. Pathfinding Algorithm and mark-based position

The pathfinding algorithm which enables hexapod robot to move from a marked world is A*. It is a search algorithm widely used in path finding and graph traversal, the process of efficiently tracing a path between points, called nodes. It is known for its performance and accuracy, and its use is widespread [30].

When A* traverses the graph, it follows the path with the lowest known cost, maintaining an ordered priority queue of alternative segments along the way. If, at any time, a segment

of the path being traversed has a higher cost than another segment of the path that has been encountered, the higher cost path segment is abandoned and the lower cost path segment is traversed instead. This process continues until the goal is reached. A* uses a best-first search and finds a least-cost path from a given initial node to a goal node (of one or more possible goals).

Due to the way the operation of this part was planned, it was decided to divide the algorithm into 3 parts:

- Map: a module which decides the location of the map where the robot is going to move
- Reasoner: a module to estimate the steps of the map where the robot has to move to achieve the goal
- Executor: a module to receive the visual current position and act accordingly by sending the necessary commands to move the robot.

Fig. 6. Distribution of marks on the ground.

The map was composed by a matrix of size NxM as it is shown in Figure 6. Each ArUco mark recognized by the camera gaves us information about its identifier, its position and rotation with respect to the camera. With these variables and knowing the position that corresponds to each mark (Figure 7), we could make the algorithm know which marks it has to pass over to reach the target we indicate. The marks were placed in the form of a matrix as this is how the camera best captured the marks and also made it easier for the pathfinding algorithm to know the position in which it was located, as well as its displacement.

The position and initial rotation of the robot and the box that the robot has to reach are initialized in the map script, once the 3 scripts are initialized, they would work as follows:

- The map indicates to the reasoner script the position and rotation of the robot.
- In addition to the target square, the reasoner calculates the route it has to follow by means of the algorithm A*, once the route is calculated, it indicates to the script that is in charge of moving the robot to which square it has to go and if it has to rotate previously or not

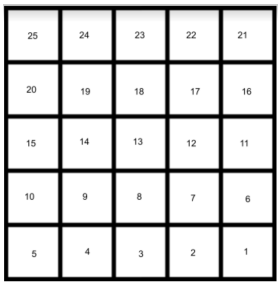

Fig. 7. Equivalence of positions and marks identifiers.

- At this moment is when the recognition of the marks comes into play, the script that is in charge of the movement consults which mark corresponds to the square it has to go, if it has to rotate, it would rotate until finding the mark and it would stop.
- Once robot reaches the mark, it communicates to the map that it has reached the position with the desired rotation and position, the map updates the position of the robot in the matrix and sends the information back to the reasoning script, and so on until the goal has been reached and the starting position has been returned to. Each and every movement involved in autonomous navigation is performed by the executing script through GET requests to the API/REST service, which must be active before starting the autonomous navigation.

IV. Results

As mentioned before, the experiments were carried out by placing the marks in the form of a matrix, more specifically, a 3x3 matrix. The experiment was run 20 times, the USR++ started at initial position which corresponds to the marker with identifier equal to 1, as shown in Figure 7, the objective was to reach the target mark and return. The target mark varied but always had to pass at least 3 marks to reach the target mark. For example, if the target mark was mark 8, I had to pass through mark 6, then mark 7, reach mark 8 and make the same way back.

The percentage of success, of the 20 executions, 17 reached the target mark correctly (85%). In the error founds: i) it did not adjust enough to the final mark (it stayed further back than necessary but finished the execution correctly) and the 2 missing ones: it lost sight of the target mark and when it moved backwards, it moved so much that it was impossible to see the mark that corresponded to it again.

One of the successful executions is shown in the following video https://youtu.be/Lo-6ARBM_mU

V. Conclusions and ongoing works

We have detailed the integration of visual servoing to a hexapod robot to perform autonomous navigation tasks. Rest services are included to provided remote controlling. In addition, we have included visual marker orientation using Aruco marks with a embedded A-star path algorithm to navigate over the map. The hexapod robot USR++

was presented in the European Researchers' Night in the University of Jaen in 2021 to show children how to compose a robot with visual capabilities using different components (https://twitter.com/epsjaen/status/1441423419573145602?s=20&t=xf6Hec0pHRja9DmTdvSOnw).

With 85 per cent of reliability in the system, the system is suitabe for educational and prototyping purposes. A possible improvement would have been to implement a routine that further developed what the robot would have to do if it did not find the target mark.

One of the big problems that has been found developing the system is the slowness with which the camera takes the images, this problem would be solved if some kind of platform was designed where the camera, instead of being in first person, would be in third person, just above the marks, that together with another mark that had the USR++ placed on top, would make a much more accurate system, this architecture Eye-to-hand architecture.

ACKNOWLEDGMENT

This publication received funding from the European Union's Horizon 2020 research and innovation programme - Pharaon Project 'Pilots for Healthy and Active Ageing' under Grant agreement no. 857188.

REFERENCES

[1] L. Atzori, A. Iera, G. Morabito, The internet of things: A survey, Comput. Netw. 54 (15) (2010) 2787–2805.

[2] M. Waibel, M. Beetz, J. Civera, R. D'Andrea, J. Elfring, D. Galvez-zLopez, K. Haussermann, R. Janssen, J. Montiel, A. Perzylo, B. Schiele, M. Tenorth, O. Zweigle, R. D. Molengraft, Roboearth, IEEE Robotics & Automation Magazine 18 (2) (2011) 69–82.

[3] Grieco, L. A., Rizzo, A., Colucci, S., Sicari, S., Piro, G., Di Paola, D., & Boggia, G. (2014). IoT-aided robotics applications: Technological implications, target domains and open issues. Computer Communications, 54, 32-47.

[4] Ding, X., Wang, Z., Rovetta, A., & Zhu, J. M. (2010). Locomotion analysis of hexapod robot. Climbing and walking robots, 30, 291-310.

[5] Land, M., Mennie, N., & Rusted, J. (1999). The roles of vision and eye movements in the control of activities of daily living. Perception, 28(11), 1311–1328.

[6] Jangir, R., Hansen, N., Ghosal, S., Jain, M., & Wang, X. (2022). Look Closer: Bridging Egocentric and Third-Person Views with Transformers for Robotic Manipulation. IEEE Robotics and Automation Letters.

[7] K. M. Abughalieh, B. H. Sababha, and N. A. Rawashdeh, "A video-based object detection and tracking system for weight sensitive UAVs," Multimedia Tools and Applications, vol. 78, no. 7, pp. 9149–9167, 2019.

[8] Chen, M. (2022). Recognition and Localization of Target Images for Robot Vision Navigation Control. Journal of Robotics, 2022.

[9] Munoz, D. J. (2006). Proceso de reconocimiento de objetos, asistido por computador (visión artificial), aplicando gases neuronales y técnicas de minería de datos. Scientia et technica, 1(30).

[10] Culcay Oñate, G. (2022). API REST para la transmisión de información y control de redes de sensores IOT (Bachelor's thesis, Universidad Técnica de Ambato. Facultad de Ingeniería en Sistemas, Electrónica e Industrial. Carrera de Ingeniería en Electrónica y Comunicaciones).

[11] Garrido-Jurado, S., Munoz-Salinas, R., Madrid-Cuevas, F. J., Marín-Jiménez, M. J. (2014). Automatic generation and detection of highly reliable fiducial markers under occlusion. Pattern Recognition, 47(6), 2280-2292.

[12] Romero-Ramirez, F. J., Muñoz-Salinas, R., & Medina-Carnicer, R. (2018). Speeded up detection of squared fiducial markers. Image and vision Computing, 76, 38-47.

[13] Shirai, Y.; Inoue, H. Guiding a robot by visual feedback in assembling tasks. Pattern Recogn. 1973, 5, 99-106.

[14] Garcia, G. J., Corrales, J. A., Pomares, J., & Torres, F. (2009). Survey of visual and force/tactile control of robots for physical interaction in Spain. Sensors, 9(12), 9689-9733.

[15] Chaumette, F.; Hutchinson, S. Visual Servoing and Visual Tracking. In Handbook of Robotics; Siciliano, B., Oussama, K., Eds.; Springer-Verlag: Berlin Heidelberg, Germany, 2008; pp. 563-583.

[16] B. Huang, M. Ye, S.-L. Lee, and G.-Z. Yang, "A vision-guided multirobot cooperation framework for learning-by-demonstration and task reproduction," International Conference on Intelligent Robots and Systems (IROS), pp. 4797–4804, 2017.

[17] Huang, H., Chai, P., Ehmke, C., Merewether, G., Dadabhoy, F., Feng, A., ... and Traverso, C. (2020). Agile mobile robotic platform for contactless vital signs monitoring. TechRxiv, to be published, doi, 10.

[18] Li, Y., & Košecka, J. (2020, May). Learning view and target invariant visual servoing for navigation. In 2020 IEEE International Conference on Robotics and Automation (ICRA) (pp. 658-664). IEEE.

[19] Nosrati, M., Karimi, R., and Hasanvand, H. A. (2012). Investigation of the *(star) search algorithms: Characteristics, methods and approaches. World Applied Programming, 2(4), 251-256.

[20] Babinec, A.; Jurisica, L.; Hubinsky, P.; Duchoň, F. Visual localization of mobile robot using artificial markers. Procedia Eng. 2014, 96, 1–9.

[21] Xing, B.; Zhu, Q.; Pan, F.; Feng, X. Marker-based multi-sensor fusion indoor localization system for micro air vehicles. Sensors 2018, 18, 1706.

[22] Zheng, J.; Bi, S.; Cao, B.; Yang, D. Visual localization of inspection robot using extended kalman filter and aruco markers. In Proceedings of the 2018 IEEE International Conference on Robotics and Biomimetics (ROBIO), Kuala Lumpur, Malaysia, 12–15 December 2018; pp. 742–747.

[23] Meng, Y.; Lin, K.J.; Peng, B.; Tsai, B.; Shih, C.S. Arpico: Using pictures to build localization service for indoor iot applications. In Proceedings of the 2018 IEEE 11th Conference on Service-Oriented Computing and Applications (SOCA), Paris, France, 20–22 November 2018; pp. 105–112.

[24] Xu, Z., Haroutunian, M., Murphy, A. J., Neasham, J., & Norman, R. (2021). An Underwater Visual Navigation Method Based on Multiple ArUco Markers. Journal of Marine Science and Engineering, 9(12), 1432.

[25] Bualat, M. G., Smith, T., Smith, E. E., Fong, T., and Wheeler, D. W. (2018). Astrobee: A new tool for ISS operations. In 2018 SpaceOps Conference (p. 2517).

[26] Simoens, P., Dragone, M., & Saffiotti, A. (2018). The Internet of Robotic Things: A review of concept, added value and applications. International Journal of Advanced Robotic Systems, 15(1), 1729881418759424.

[27] Roy Chowdhury, A. (2017). IoT and Robotics: a synergy. PeerJ Preprints, 5, e2760v1.

[28] Neumann, A., Laranjeiro, N., and Bernardino, J. (2018). An analysis of public REST web service APIs. IEEE Transactions on Services Computing.

[29] Gómez Díaz, M. (2019a, abril). Control de un robot hexápodo. Jaén: Universidad de Jaén.

[30] Rana, K., & Zaveri, M. (2011). A-star algorithm for energy efficient routing in wireless sensor network. Trends in Network and Communications, 232-241.