23rd ABCM International Congress of Mechanical Engineering
December 6-11, 2015, Rio de Janeiro, RJ, Brazil

DEVELOPMENT OF AN AUTONOMOUS HEXAPOD ROBOT WITH DIGITAL IMAGE PROCESSING IMPLEMENTED ON RASPBERRY PI

Rodrigo Barbosa Letang
Ricardo Naoki Mori
Alexandre Brincalepe Campo
Federal Institute of Technology in Sao Paulo, Street Pedro Vicente, 625, São Paulo (SP), Brazil
rodrigoletang@gmail.com; ricardo_mori@ifsp.edu.br; brincalepe@gmail.com

Abstract. *Currently many of robotic systems are developed with some kind of inspiration in nature, seeking to get flexibility and the efficiency of biological systems. An example are the hexapod robots, six-legged robots which movements try to imitate the movements of the insects, walking with two groups of three legs alternately, providing stability to the robot, flexibility in movement and mobility on irregular surfaces. With these attributes the hexapod robots can be used in exploration of irregular terrains, inhospitable places or places with difficult access to humans. This project involves the development in an embedded system of an autonomous hexapod robot (Beetle-I) with digital image processing implemented on Raspberry Pi, to studies in the areas of autonomous robotic systems with legged locomotion and robotic vision. The scientific contribution of this paper consists in an integration of a robotic system and an embedded system of digital image processing, programmed in high level language using Matlab/Simulink tools. The robot has three degrees of freedom in each one of the six legs, is equipped with a camera to capture images and also uses a distance sensor that allow the robot to detect obstacles. The movements of each "leg" of the robot are performed by calculating the inverse kinematics for each point of a discretised trajectory, determining the angles required in each joint axes, which in turn is converted into PWM control signal for servomotors. This control was implemented in a Digital Signal Controller (DSC), a hybrid device that combines the functionalities of a microcontroller with the processing capacity of a Digital Signal Processor (DSP). The Hexapod Robot Beetle-I is able to recognize the presence of a green tennis ball on a neutral scenario, where doesn't occurs the object camouflage. The digital image processing and the distance detection was implemented on a Raspberry Pi, an integrated computer on a single board with a Raspbian Linux system. It was also planned a connection between the Raspberry Pi and the Digital Signal Controller (DSC) to enable a future implementation of a robot navigation control on the Raspberry Pi.*

Keywords: *Hexapod Robot, Digital Image Processing, Raspberry Pi, Inverse Kinematics, DSC.*

1. INTRODUCTION

According to SILVA (2005), the robots with legged locomotion have a higher mobility than the wheeled vehicles in natural terrains, since these robots may use isolated footholds for each foot, as opposed to the wheeled systems, which need a continuous support surface. So, these robots can walk on irregular terrains, varying the configuration of the legs to adapt to the irregularities of the surface. On the other hand, the legged locomotion system also presents some limitations in relation to the wheeled locomotion system, such as lower speed and greater complexity in the construction and in the control algorithms.

The use of multiple degrees of freedom at the joints of the legs allows the legged robots to change their movement direction without slippage. Additionally, it is possible to change the height from the ground, introducing a damping and a decoupling between the terrain irregularities and the body of the robot. Aiming to decrease the weight of the robot and the required number of actuators, it is desirable that the robot has the lowest possible number of legs. On the other hand, at least four legs are needed to the robot perform a locomotion maintaining the static stability, always having three footholds while a leg moves (SILVA, 2005).

One of the most common legged robots is the hexapod robot, which have six legs and try to imitate the movements of insects, class of beings with the greatest diversity on the planet. The most common walking way of the hexapod robots is to walk on two groups of three legs alternately (LETANG *et al.*, 2008). This walking way is called tripod gait, in which three legs support the weight of the robot while the other three move to the next position, where they will in turn support the robot to the first three legs starting the movement (LETANG *et al.*, 2010).

The legged robots have many applications in exploring remote locations and hostile or hazardous environments. Among them may be mentioned: space exploration; volcanoes; seabed exploration; nuclear power plants; places with high levels of radiation; mineral exploration; disaster areas; search and rescue operations (SILVA e MACHADO, 2001).

In autonomous mobile robotic systems are commons the abilities to navigate from a known position to a new location, to avoid obstacles and to position on a task. These abilities are possible with the use of a sensing system which acquire data that describing the environment and send them to the robot processing system (JÁCOBO, 2001).

Some autonomous mobile robots are specially designed to study the dynamic control of robots, using force sensors attached to the structure of the members (BACHEGA *et al.*, 2012). Others are designed to study the kinematic control of robots. These use other types of sensors for interacting with the environment, such as distance sensors and computer vision systems (LETANG *et al.*, 2008).

The distance sensors are devices that measure a distance between a reference point and objects. These sensors are used in robot navigation and obstacle avoidance. The distance sensors most commonly used in autonomous mobile robots are the ultrasonic sensors, which gets the distance between the sensor and the object by the time the ultrasonic signal takes to intercept the object and returns to the sensor (JÁCOBO, 2001).

About the computer vision, according to FAIRHURST (1988), it can be divided into two areas that relate: Image Processing and Pattern Recognition. According to GOMES e VELHO (1994), the Image Processing is used to emphasize some characteristic like the outline of objects and obtain geometric, topological or physical information about the scene. The Pattern Recognition is used to extract data from the pre-processed image, like find similar images, identify an object or determine the type of texture of an object (CAVANI, 2004).

The main types of camera systems for computer vision used by robots are: monocular vision (one camera), stereo vision (two or more cameras) and omnidirectional vision (conical, spherical or hyperbolic mirrors combined with one camera). With the improvement of methods and techniques that make intelligent use of the information provided, the monocular vision system has proven to be a viable option (COUTO, 2012).

This work aims a development of an autonomous mobile robotic system with legged locomotion, hexapod type, with three degrees of freedom in each leg, tripod gait walking mode, with ultrasonic distance sensor and monocular vision sensor, designed for studies in the fields of kinematic control of autonomous robots and robotic vision, using Matlab/Simulink image processing tools embedded in a Raspberry Pi.

The Hexapod Robot (Beetle-I) must be able to detect obstacles ahead and recognize the presence of a green tennis ball on a neutral scenario, where doesn't occurs the object camouflage. A connection between the Raspberry Pi and the Digital Signal Controller (DSC) should be designed to enable a future implementation of a navigation control of the robot on Raspberry Pi.

2. DEVELOPMENT

The development of this work can be divided into three parts: Robot Development, Vision System Development and Distance Sensing Development.

2.1 Robot Development

The development of this robot is based on the previous works: "Desenvolvimento de um Robô Hexápode" (LETANG et al., 2008) and "Kinematics Open Loop Control of Hexapod Robot with an Embedded Digital Signal Controller (DSC)" (LETANG *et al.*, 2010).

To avoid an unwanted bending in the legs of the robot while walking, as occurred in the previous robot, it would be necessary to design a more robust structure using aluminum. However, as the mechanical design and the manufacture of parts of this structure are not in the scope of this work, it was decided to use the Hexapod Kit MSR-H01 of Micromagic Systems company, which consisting of 26 aluminum parts.

The actuators used in this project are servo motors that have a big advantage because of having an internal closed loop that keeps the servo motor axis in the angular position corresponding to the input signal generated by Pulse Width Modulation (PWM).

The servo motors used was the servo motors indicated by the manufacturer of the hexapod kit which, apart from supporting the weight of the structure, are mechanically compatible with the dimensions of the robot parts. In the robot's body were used 12 servo motors Hitec HS-645MG with metal gears and torque of 0.47 Nm, and 6 servo motors Hitec HS-225MG with metal gears and torque of 0.94 Nm.

In these servo motors the PWM input signal should have a period of 20ms and the pulse width range is between 1ms and 2ms, which corresponds to an angular range of approximately 100°. However, the mechanical limitations of the servo motor allow an angular range of 200° using pulse widths beyond the ideal range, between 0.5ms and 2.5ms.

This project uses a Digital Signal Controller (DSC), a hybrid device that combines the processing capacity of a DSP (Digital Signal Processor) with the functionalities of a microcontroller, that has various peripherals. The DSC used is the MC56F8025 of Freescale, which has a processing capacity of 32MIPS (million of instructions per second), 32KB Flash memory, 4KB RAM memory and 35 GPIOs (digital input/output).

The battery used in this project was a Nickel Metal Hydride (Ni-MH) battery with nominal voltage 6V and charge capacity 4,3Ah. This battery has been recommended by the manufacturer of hexapod kit for the ability to supply the power of the servo motors and the dimensions compatible with the physical robot structure.

The robot eletronic circuit was set to minimize energy expenditure to get the highest possible autonomy of the battery and aiming to minimize the size of the printed circuit board because the internal space available on this robot is quite restricted. The circuit contains: a regulated power supply at 3.3V for the DSC; a regulated power supply at 5V for the Raspberry Pi; the DSC MC56F8025; 20 outlputs to the servo motors; one JTAG communication interface for programming and debugging; a connector attached to all DSC free pins to predict a future connection with the Raspberry Pi.

The kinematic modeling of the robot can be divided into two steps: the forward kinematics and the inverse kinematics. In the forward kinematics, knowing the geometry of the robot legs, it is possible to develop a matrix mathematical model that, with the angles of each joint, calculates the position of the end of the leg that touches the ground (known as the terminal element). In the inverse kinematics, the forward kinematic matrix model is manipulated and results in a model that, with a future position of the terminal element, calculates the combination of the joint angles that would lead the terminal element to the new position. In an extrapolation, it is possible to calculate trajectories to move the robot.

For the development of the mathematical model of the forward kinematics was used theDenavit-Hartemberg method. In this method, each joint is assigned to a cartesian coordinate system (x_i, y_i, z_i), which are related by Denavit-Hartemberg parameters ($\boldsymbol{a_i}$, $\boldsymbol{\alpha_i}$, $\boldsymbol{d_i}$, $\boldsymbol{\theta_i}$) in order to reference the position of the terminal element in the robot geometric center. In the Figure 1 it is possible to observe the design of the partially assembled robot, the coordinate systems assigned to each vertex, and the definition of the displacement vector, which is the vector between the current point and the future point of the terminal element. The index of Denavit-Hartemberg parameters starts at zero in the center of the robot and is incremented on each joint toward the terminal element.

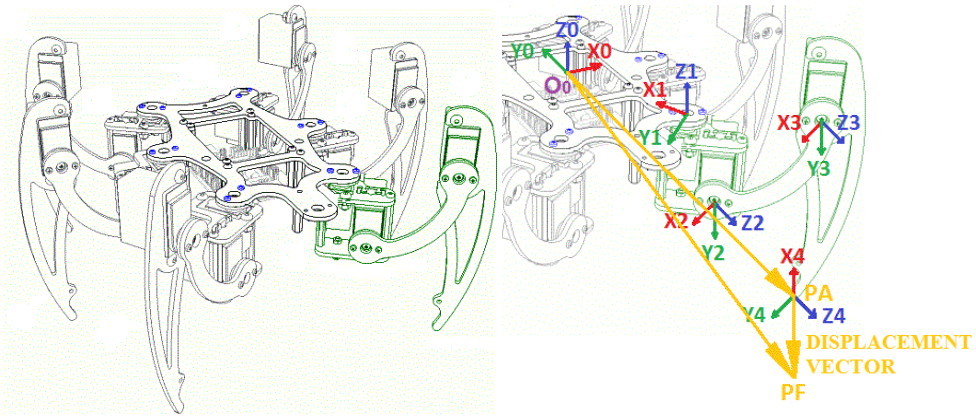

Figure 1 – Hexapod Robot partially assembled, coordinate systems and displacement vector.

The Denavit-Hartemberg parameters represent:

$\boldsymbol{a_i}$ – distance between the z_{i-1} and z_i axes measured over the x_i axis;
$\boldsymbol{\alpha_i}$ – angle between the z_{i-1} and z_i axes measured in a normal plane to x_i axis. The direction of angle is given by the right hand rule in the x_i axis;
$\boldsymbol{d_i}$ – distance between the origin O_{i-1} and the x_i axis intersection with the z_{i-1} axis measured over the z_{i-1} axis;
$\boldsymbol{\theta_i}$ – angle between the x_{i-1} and x_i axes measured in a normal plane to z_{i-1} axis. The direction of angle is given by the right hand rule in the z_{i-1} axis.

The application of this method in a structure such as the legs of the robot, where all the joints are revolution joints, resulting in constant parameters a_i, α_i, d_i e θ_0 and just θ_2, θ_3 e θ_4 variables, which are the angles of the joints.

The parameters of each joint are used to define Homogeneous Transformation Matrix (Fig. 2a) that represents the transformation of the coordinate system *i* to the coordinate system *i-1*. The four homogeneous transformation matrices are multiplied to generate the Global Homogeneous Transformation Matrix (Fig. 2b) which provides the forward kinematics of a robot leg and also represents the transformation of the terminal element coordinate system to the base coordinate system.

$$A_i = \begin{bmatrix} \cos\theta_i & -\cos\alpha_i\sin\theta_i & \sin\alpha_i\sin\theta_i & a_i\cos\theta_i \\ \sin\theta_i & \cos\alpha_i\cos\theta_i & -\sin\alpha_i\cos\theta_i & a_i\sin\theta_i \\ 0 & \sin\alpha_i & \cos\alpha_i & d_i \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

$$A_{Global} = A_1 * A_2 * A_3 * A_4$$

Rotation Matrix — Vector (Origin to Origin)

$$A_{Global} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & O_x \\ r_{21} & r_{22} & r_{23} & O_y \\ r_{31} & r_{32} & r_{33} & O_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Scale

Figure 2 - Homogeneous Transformation Matrix (a) and Global Homogeneous Transformation Matrix (b).

So, with the angles of each joint (θ_2, θ_3 e θ_4), it is possible to calculate the coordinates *Ox*, *Oy* e *Oz*, that determine the position of the terminal element in the base coordinate system, which is located in the center of the robot. The Rotation Matrix shown in Fig. 4 represents the rotation of the terminal element coordinate system in relation to the base coordinate system and has not been used in this project.

In the inverse kinematics, the vector composed by the terms *Ox*, *Oy* e *Oz* of the global homogeneous transformation matrix is manipulated to obtain the joint angles required for the terminal element moves to a certain position in the base coordinate system. The Taylor series, truncated at the first derivative, results in the Eq. (1):

$$FP(k) = CP(k) + \frac{d[CP(k)]}{dk} \cdot \Delta k \quad (1)$$

The term *PF* is the projection of the future point on the base coordinate system and is obtained by multiplying the future point in the terminal element coordinate system by the global homogeneous transformation matrix. The term *PA* is the current point in the base coordinate system and is obtained by the vector composed of the terms *Ox*, *Oy* e *Oz*. Because it is a three-dimensional function, the derivative is replaced by a jacobian matrix $J(\theta_{k-1})$ which is formed by the first order partial derivatives of the vector function defined by *Ox*, *Oy* e *Oz*, in relation to θ_2, θ_3 e θ_4. The term Δk is the variation of the angles θ_2, θ_3 e θ_4 and also the variable of interest of the equation. Replacing the terms and isolating the variable of interest, results in the equation of inverse kinematics of a hexapod robot leg:

$$[\Delta\theta_i] = [J(\theta_{k-1})^{-1}] . [FP(\theta_i) - CP(\theta_{i-1})] \quad (2)$$

Finally the joint angles required for the terminal element moves to the future point are obtained performing the addition $\theta_{i-1} + \Delta\theta_i$. The complete process can be visualized in Fig. 3.

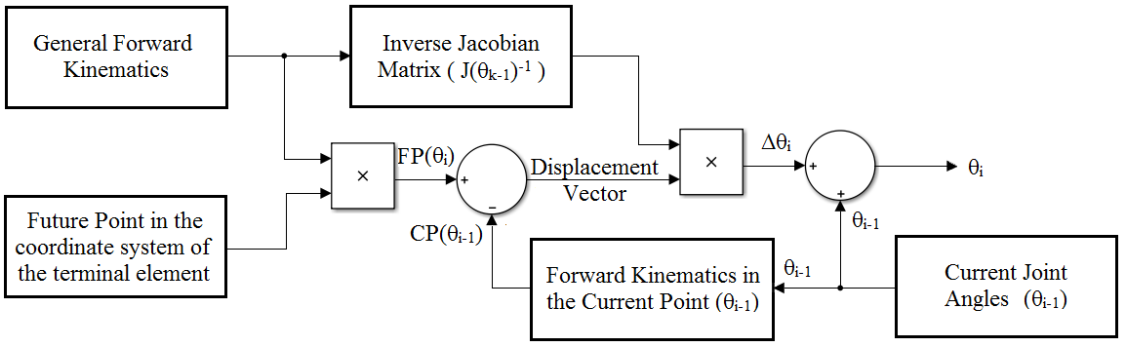

Figure 3 – Inverse Kinematics.

This process is performed at each point of a discretized parabolic trajectory of the robot leg. The use of a parabolic trajectory was set because it is one of the most used trajectory formats in legged robots. It was developed a program in Matlab where is possible to simulate the trajectory that a leg will perform to execute one step. This program was rewritten in C language to be incorporated into the DSC.

2.2 Vision System Development

The computational cost of an embedded vision system requires a high processing capacity platform. In this project the digital image processing was implemented on a Raspberry Pi Model B (Fig. 4) with a Raspbian Linux operating system installed on a 8GB SD card. The Raspberry Pi Model B is an integrated computer on a single board with a 700MHz ARM11 processor, 512MB RAM and 8 GPIOs (digital input/output).

Figure 4 – Raspberry Pi (RASPBERRY PI FOUNDATION, 2014).

The program was developed using a Simulink Support for Raspberry Pi. With it is possible to generate a program in Simulink, transfer it via network to the Raspberry Pi using the SSH protocol and run it. Once the program has been downloaded is possible to run it locally without the supervision of the Simulink. The support also allows to incorporate a program written in Matlab into a Simulink block.

The image acquisition uses a V4L2 Video Capture Simulink block that imposes some restrictions to the vision sensor, which should have support to a USB video device class called UVC and to the YUYV color representation format. According to the list of cameras tested by Mathworks, it was decided to use the webcam Logitech C525.

The recognition of the presence of a green tennis ball on a neutral scenario, where doesn't occurs the object camouflage, was developed in two different ways: Green Color Tone Recognition and Image Pattern Recognition. In both cases the recognition was implemented in a program developed in Matlab and incorporated into a Simulink block.

In the Green Color Tone Recognition the algorithm captures the RGB components of the camera signal and calculates the green color tone of each point of the image using the equation $G - R/2 - B/2$ (MATHWORKS, 2011). Then it checks if there are points in the image with color tone in a certain range. In affirmative case, it lights a green LED connected to the Raspberry Pi GPIO22 pin. The program Simulink blocks is shown in Fig. 5.

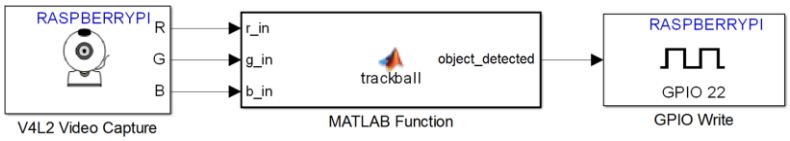

Figure 5 – Green Color Tone Recognition.

In the Image Pattern Recognition, the algorithm compares characteristic points of the object and the scene in search for similarities, filters isolated equivalent points and obtains the geometric transformation between groups of points. The object detection is determined by the number of equivalent points. The program was implemented in Simulink, first developed using a JPEG file with the image of the scene (Fig. 6) and later using the camera image (Fig. 7).

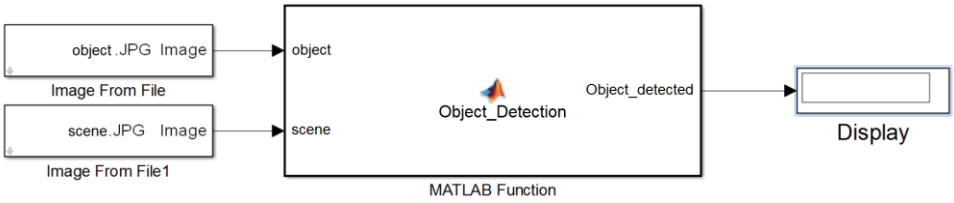

Figure 6 – Image Pattern Recognition with JPEG files.

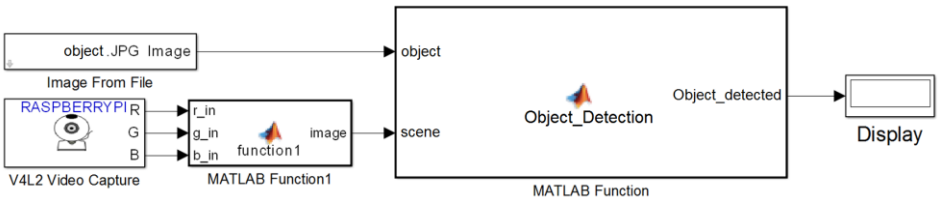

Figure 7 – Image Pattern Recognition with the camera.

The program uses Matlab instruction based on Speeded-up Robust Features (SURF) algorithm, which is a descriptor detector method based on scale space. The algorithm search for compatible reference points between the images. The amount of these points can be used as a criterion for evaluating the similarity between the scenes. This method generates good signatures of reference points in the image, relatively resistant to the prospect changes, rotation, scale, lighting, occlusion and noise.

2.3 Distance Sensing Development

For the distance sensing was used the ultrasonic module HC-SR04, which has the transmitter, the receiver and the control circuitry on a single board. The module works by emitting an ultrasonic pulse and measuring the time the pulse takes to reflect an obstacle and return to the receiver. With this time is possible to calculate the distance between the sensor and the obstacle. The module is capable of sensing distances between 20 mm and 4 meters. The program that interacts with the ultrasonic module and calculates the distance to the obstacle has been developed in Python language and implemented directly on Raspberry Pi using GPIOs. To initiate the process, the program sends a Trigger signal to the module, that sends an ultrasonic signal to the transmitter and waits for the return of the signal to the receiver. Then the module generates a pulse on the Echo pin that represents the time the signal took to return to the receiver. The duration of this time is accounted for by the program that divides the time by two and multiplies it by the sound speed (340m/s). As the maximum distance is 4 meters, the maximum length of the Echo pulse is 23,5ms. If the sensor don't intercept any obstacle, it generates a 38ms pulse (HAWKINS, 2012). Finally if an obstacle is detected with less than 20 cm distance the algorithm lights a red LED connected to a Raspberry Pi GPIO pin.

3. RESULTS

The results of this work can be divided into three parts: Robot Development, Vision System Development and Distance Sensing Development.

3.1 Robot Development

The robot physical structure was built with servo motors, battery, printed circuit board, Raspberry Pi, camera and ultrasonic sensor. The Beetle-I robot completely assembled (Fig. 8) weighs 2.3kg and its dimensions in standard standing position are 17.5 cm high, 31,5cm wide and 33,5cm long. In the practical tests the battery lasted 85 minutes supplying the robot in standard standing position.

Figure 8 – Beetle-I robot.

Simulations were performed to compare the trajectory that the robot leg perform using the results of calculations of the forward and inverse kinematics, with a theoretical parabolic trajectory. For this it was used the forward and inverse kinematics calculation program developed in Matlab. The simulation plotted (Fig. 9) the theoretical parabolic trajectory in red and the practical trajectory in blue. The error module between the two parables, point by point, reached a maximum of 0,2155mm and was considered satisfactory.

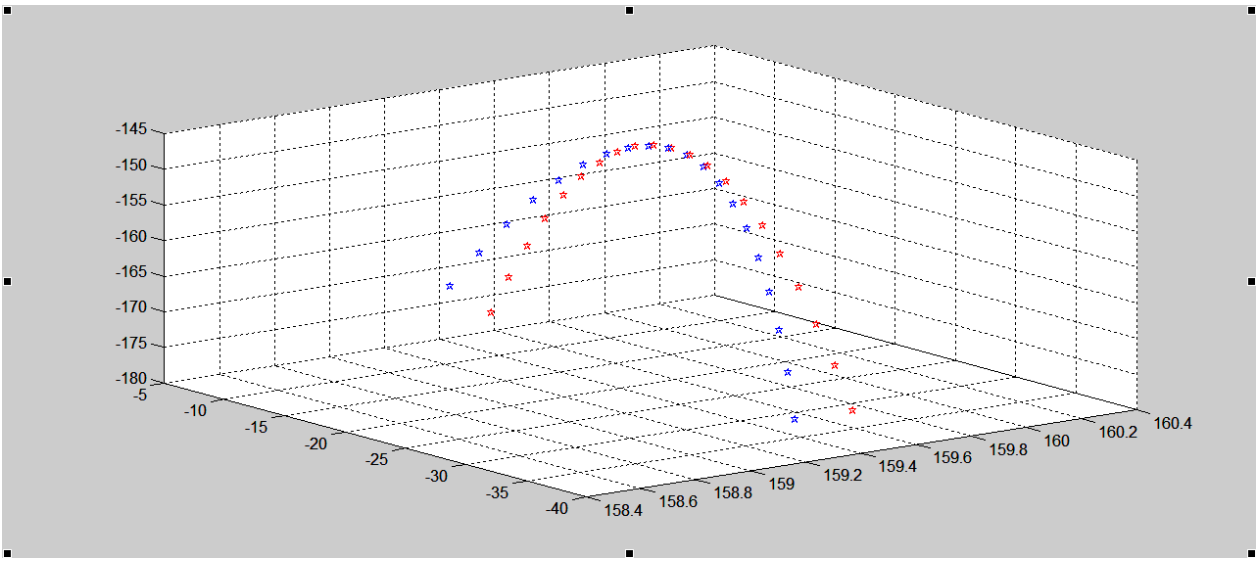

Figure 9 – Graph in mm of the comparison between the theoretical trajectory (red) and practical trajectory (blue).

3.2 Vision System Development

In the method of Green Color Tone Recognition was used the program described and a green LED connected to the Raspberry Pi that lights up when the algorithm identifies a green color tone close to the tennis ball. The program was compiled in Simulink in standalone mode to run directly on the Raspberry Pi and worked normally. In practical tests the algorithm could correctly identify the tennis ball in various scenarios since there were no other object with similar color tone. In Table 1 is possible to see what green color tone the algorithm recognizes as the green tennis ball.

Table 1 – Green Color Tone Recognition

Colors	Object Recognized
	No
	No
	No
	No
	No
	Yes

In the method Image Pattern Recognition was used the program described for this method and first tested in "Normal" simulation mode in Simulink, running only on the computer and using a JPEG file with the image of the scene. The tests showed satisfactory results, achieving to recognize the tennis ball and a more complex object.

After converting the images of the tennis ball and the scene for grayscale, the program identified some equivalent characteristic points between the two pictures. After filtering isolated points the program generated as output the Fig. 10, showing the equivalent points, the detected object and its inclination on the scene. The same procedure was performed using a more complex object and the program generated as output the Fig. 11.

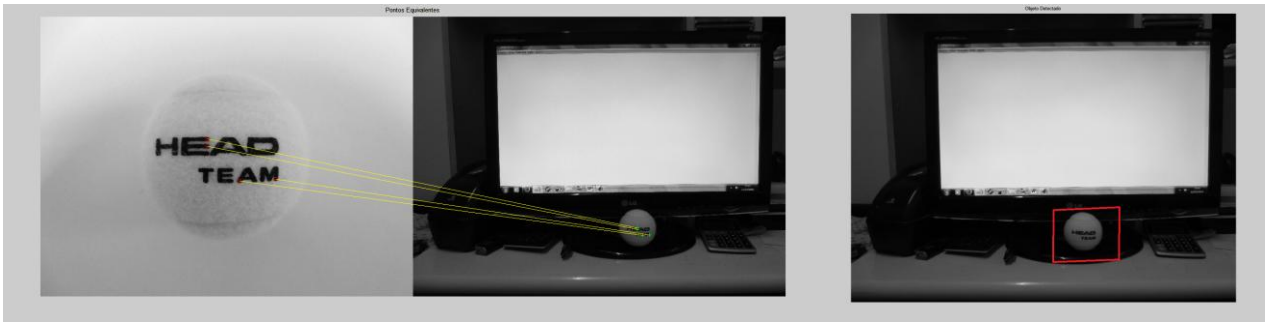

Figure 10 – Identification of equivalent characteristic points of a tennis ball using SURF algorithm.

Figure 11 – Identification of equivalent characteristic points of a complex object using SURF algorithm.

Then it made a program compilation attempt in standalone mode to run directly on the Raspberry Pi and using the camera image. This attempt showed some errors reporting that some functions like rgb2gray and mainly functions based on SURF algorithm (detectSURFFeatures, matchFeatures, extractFeatures and estimateGeometricTransform) used in the program of a Simulink function block are not available for code generation in standalone mode.

As the image pattern recognition program depends on the outputs of these functions to run the SURF algorithm, the execution of the program on the Raspberry Pi was impossible by the incompatibility of these functions with the Simulink Support for Raspberry Pi in standalone mode. A possible solution would be to rewrite the program of these functions, but the complexity of the algorithm and the absence of detailed documentation of these Matlab functions hampered the process.

3.3 Distance Sensing Development

Measurements were performed with the objective of comparing the distance measured by the sensor with the actual distance between the ultrasonic sensor coupled to the head of the robot and an obstacle. For this it was used the program described with a slight modification to print on the screen the measured distance. The result of the comparison of program measurements with the actual distances presented a maximum error of +/- 4%.

4. CONCLUSIONS

The simulations to compare the trajectory that a robot leg performs using the calculations of forward and inverse kinematics with a theoretical parabolic trajectory were satisfactory, presenting a relative maximum error 0,2155mm.

The method of green color tone recognition was able to recognize the presence of a tennis ball in different scenarios and was efficient in distinguishing the green color tone of the tennis ball from the other tones. The method presented a limitation in scenarios with other objects with green color tone very close to the tennis ball color, creating fake identifications. This limitation was expected due to the simplicity of the method.

The method of image pattern recognition using the SURF algorithm was able to recognize both the presence of a tennis ball as the presence of a more complex object when tested in "Normal" simulation mode in Simulink, running only on the computer and using a "JPEG" file with the image of the scene. But it was not possible to implement this method in embedded way using the camera image due to limitations of the Simulink Support for Raspberry Pi, which can not perform code generation in standalone mode when certain features are present in the program.

The measurements of the distance sensing presented a maximum relative error of +/- 4% which was considered acceptable for the application, that is to detect obstacles with less than 20 cm distance.

With these results it was possible to achieve the objectives of the work implementing the studies in the areas of kinematic control of autonomous robots and robotic vision, in order to perform a trajectory with the robot leg right next to the idealized trajectory, using the techniques of forward and inverse kinematics, to recognize the presence of a green tennis ball in a neutral scenario where doesn't occurs the object camouflage, and to detect the presence of obstacles in front of the robot with acceptable accuracy. However, the use of Matlab/Simulink image processing tools embedded in a Raspberry Pi had some limitations that were presented in this paper.

In addition, was designed a connection between the Raspberry Pi and the Digital Signal Controller (DSC) to enable a future implementation of a navigation control of the robot on Raspberry Pi, which would integrate the robot movement control system of the DSC with the recognition of green tennis ball and the distance sensing of the Raspberry Pi. In these future work the Raspberry Pi would communicate with the DSC through 6 GPIOs that would determine the movement to be executed: forward, backward, left, right, clockwise rotation or counterclockwise rotation.

Another future work that could be developed would be the image pattern recognition without using Simulink Support for Raspberry Pi. An alternative would be to implement directly on Raspberry Pi programming in C++ or Python language and using the OpenCV library that has several digital image processing functions.

5. REFERENCES

BACHEGA, R. P.; CAMPO, A. B. and PIRES, R., 2012. "Hardware Configuration of Hexapod Robot to Force Feedback Control Development". *Proceedings of the 44th IEEE Southeastern Symposium on System Theory*. Jacksonville, Florida.

CAVANI, F. A., 2004. *Sistema de Visão Omnidirecional para Robô Móvel*. Bachelor Monograph, Universidade Estadual Paulista (UNESP), São José do Rio Preto, São Paulo.

COUTO, L. N., 2012. *Sistema para Localização Robótica de Veículos Autônomos Baseado em Visão Computacional por Pontos de Referência*. Masters Dissertation, Universidade de São Paulo (USP), São Carlos, São Paulo.

FAIRHURST, M. C., 1988. *Computer Vision for Robotic Systems: An Introduction*. Prentice Hall.

GOMES, J. and VELHO, L., 1994. *Computação Gráfica: Imagem*. Instituto de Matemática Pura e Aplicada (IMPA).

HAWKINS, M., 2012. "Ultrasonic Distance Measurement Using Python – Part 1". 08 Marc. 2015. <http://www.raspberrypi-spy.co.uk/2012/12/ultrasonic-distance-measurement-using-python-part-1/>.

JÁCOBO, J. E. A., 2001. *Desenvolvimento de um Robô Autônomo Móvel Versátil utilizando Arquitetura Subsumption*. Masters Dissertation, Universidade Estadual de Campinas (UNICAMP), Campinas, São Paulo.

LETANG, R. B.; CARVALHO, R. C. F. P.; MORAES, W. M.; SCHNEIATER, R.; TOTAKI, M. K.; CAMPO, A. B., 2008. *Desenvolvimento de um Robô Hexápode*. Graduation Article, Universidade São Judas Tadeu, São Paulo.

LETANG, R. B.; CARVALHO, R. C. F. P.; MORAES, W. M.; SCHNEIATER, R.; TOTAKI, M. K.; CAMPO, A. B., 2010. "Kinematics Open Loop Control of Hexapod Robot with an Embedded Digital Signal Controller (DSC)". *Proceedings of the IEEE International Symposium on Industrial Electronics (ISIE2010)*. Bari, Italy.

MATHWORKS, 2011. "MATLAB Examples". 08 Marc. 2015. <http://www.mathworks.com/examples/matlab/3962-find-green-object>.

RASPBERRY PI FOUNDATION, 2014. "Raspberry Pi".11 Aug. 2014. <http://www.raspberrypi.org/>.

SILVA, M. and MACHADO, J. A. T, 2001. "Sistemas Robóticos de Locomoção - Estado da Arte". *Instituto Ingenium, Porto*, 2ª Série, n. 62.

SILVA, R. D. S. E., 2005. *Locomoção de Robôs Caminhantes Inspirada em Sistemas Biológicos*. Graduation Monograph, Universidade Federal de Ouro Preto (UFOP), Ouro Preto, Minas Gerais.

6. RESPONSIBILITY NOTICE

The authors are the only responsible for the printed material included in this paper.