Research, Society and Development, v. 11, n. 2, e38211225857, 2022
(CC BY 4.0) | ISSN 2525-3409 | DOI: http://dx.doi.org/10.33448/rsd-v11i2.25857

PI controller implementation for the two wheels of a differential robot using NI-MyRio

Implementação do Controlador PI para as duas rodas de um robô móvel diferencial com NI-MyRio

Diseño de un controlador PI para las ruedas de un robot móvil diferencial utilizando NI-MyRio

Received: 01/18/2022 | Reviewed: 01/22/2022 | Accept: 01/27/2022 | Published: 01/28/2022

Ronaldo do Amaral Oliveira
ORCID: https://orcid.org/0000-0003-4781-088X
Instituto Federal de Educação Ciência e Tecnologia do Espírito Santo, Brazil
E-mail: ronaldo.oliveira@ifes.edu.br
Marco Antonio de Souza Leite Cuadros
ORCID: https://orcid.org/0000-0003-4191-1794
Instituto Federal de Educação Ciência e Tecnologia do Espírito Santo, Brazil
E-mail: marcoantonio@ifes.edu.br
César Silva Xavier
ORCID: https://orcid.org/0000-0002-9954-6614
Instituto Federal de Educação Ciência e Tecnologia do Espírito Santo, Brazil
E-mail: cesarbio2011@gmail.com
Carlos Torturella Valadão
ORCID: https://orcid.org/0000-0002-0765-7094
Instituto Federal de Educação Ciência e Tecnologia do Espírito Santo, Brazil
E-mail: carlostvaladao@gmail.com

Abstract
Introduction. Computational power improvement throughout the time combined with the overall technology advancement has allowed the development and use of robotics for several applications, such as supervision in hazardous places, transportation, vigilance, tourism guiding and, cleaning, among others. The mobile robotics field industry is not yet visible in Brazil, since there is no expressive national manufacturer of a platform for robotics development and programming. Objective. Thus, this article aims to show the development of a mobile robot to be used in internal and external environments for didactic purposes and as a development platform. Methodology. The development of a mobile robot includes hardware and software design, and, in this last group, there are the speed controllers, which are an important part of the robot design and building. In our article, it is implemented a PI controller in an embedded system of National Instruments, called NI-MyRio. This system is programmed in LabVIEW and embeds speed control for the wheels, besides the encoders reading. Additionally, it is connected to an Ethernet network, allowing supervision and control from one or more computers in the same network. Results. It was possible to model the wheels and to configure the PI controller by using such models and the internal control model method. In the experiments, it was possible to prove the functionality of the controllers satisfactorily. Conclusion. In conclusion, using the methodology described above, it was possible to model, build, and evaluate the robot and the controller, fulfilling the project requirements.
Keywords: Mobile Robotics; PI Controller; PID; LabVIEW.

Resumo
Introdução. Com o aumento da capacidade de processamento e o desenvolvimento da tecnologia, a robótica móvel vem sendo usada em diferentes aplicações como na supervisão em locais perigosos, transporte, vigilância, guia de visitantes, limpeza, entre outros. O crescimento desta área ainda não é visível no Brasil, sendo que não existem grandes fabricantes de robôs para desenvolvimento. Objetivo. Desta forma, este artigo visa mostrar o desenvolvimento de um robô móvel para uso em ambientes interno e externo para fins didáticos e como plataforma de desenvolvimento. Metodologia. O desenvolvimento de um robô móvel envolve elementos de hardware e software, incluindo, nesta última categoria, os controladores de velocidade que são partes importantes na construção de robôs móveis, os quais serão implementados com o controlador PI no sistema embarcado da National Instruments, chamado de NI-MyRio. Este sistema é programado no LabVIEW e embarca os controladores de velocidade das rodas, além de realizar a leitura dos encoders do robô. O NI-MyRio está conectado a uma rede Ethernet, o que permite realizar a supervisão dos procedimentos através de um ou mais computadores. Resultados. Foi possível realizar a modelagem matemática das rodas e, a partir destes modelos, configurar o controlador PI utilizando o método do modelo do controle interno. Dentro dos experimentos, foi possível comprovar a funcionalidade dos controladores de forma

satisfatória. Conclusão. Desta maneira, pode-se concluir que foi possível a construção e modelagem matemática do robô e do controlador a fim de que tivessem seus requerimentos de projeto satisfeitos.
Palavras-chave: Robótica móvel; Controlador PI; PID; LabVIEW.

Resumen
Introducción. Con el aumento de la capacidad de procesamiento y el desarrollo de la tecnología, la robótica móvil ha sido utilizada en diferentes aplicaciones, como en la supervisión de locales peligrosos, transporte, vigilancia, guía de turistas, limpieza y otros. El crecimiento de esta área todavía no es visible en Brasil, pues no hay grandes fabricantes de plataforma para desarrollo y programación de robots. Objetivo. Este artículo muestra el desarrollo de un robot móvil para uso en ambientes internos y externos para fines didácticos y como una plataforma de desarrollo. Metodología. El desarrollo de un robot móvil implica en elementos de hardware y software, y, en esta última categoría, los controladores de velocidad son partes importantes en la construcción de robot móviles, los cuales son implementados con un controlador PI embebido en un sistema NI-MyRio, de National Instruments. Este sistema es programado em LabVIEW y tiene los controladores de velocidad de las ruedas y, además, puede realizar las lecturas del encoder del robot. El sistema de NI-MyRio está conectado a una red Ethernet, en la cual computadoras conectadas a la misma pueden supervisarlo y controlarlo. Resultados. Se hizo el modelaje matemático de las ruedas y, por medio de estos modelos, se configuró el controlador PI utilizando el método del modelo del control interno. Dentro de los experimentos, se comprobó la funcionalidad de los controladores de forma satisfactoria. Conclusión. Así, se concluye que se logró encontrar el modelo matemático del robot y controlador para que atendiesen a los requisitos del proyecto.
Palabras clave: Robótica Móvil; Controlador PI; PID; LabVIEW.

1. Introduction

In the last decades, robotics has gained attention in industrial applications, aerospace, and security fields, and even accompanying elderlies (Souza, 2005). Recently, the number of mobile robots increase considerably in industry, and they perform several tasks, such as working in warehouses and controlling the inventory (Faisal et al., 2013). The control systems are present in industrial processes and home appliances, besides being an important part of communication, aviation, logistics, and robotics (Manyika et al., 2013) e (Simoens et al., 2018).

A differential robot is a small autonomous vehicle with differential traction, which means that they have two independent motor wheels, as shown in Figure 1. The motor can be connected to the wheels using different transmission rates, and the terrain can also have an impact on the performance of the wheels, since it may be responsible for slipping as in a sandy or muddy place (Ben-Ari & Mondada, 2018). The differential traction has some advantages, such as the simplicity of having just two motors without additional components to determine the direction, which is defined by the behavior of the motors combined (Y.Abdalla & I. Hamzah, 2013). An autonomous robot should execute intrinsic maneuvers with simple movements. Therefore, the differential transmission is preferred, since it can easily spin into any direction and then, move linearly in that direction (Ben-Ari & Mondada, 2018).

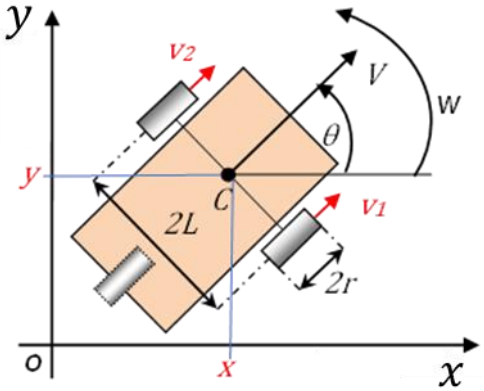
The application of the PID controller (Proportional-Integral-Derivative) based on the difference among the reference value and the response of the system has appeared in the 1940s, and, since then, it has become one of the standards of the control strategies in the processes in both academy and industry (Åström, 2002; Astrom & HÄgglund, 2006). In our article, the PI will be implemented in the embedded system NI-MyRio, which is programmed in LabVIEW. The control system of our robot requires fast and precise positioning, which is only possible when the controllers have a good performance, independently of the complexity (Sharma et al., 2017). It will be applied two PI controllers, one for each wheel, to control the speed of the wheels of a differential robot. To allow an automatic speed control of the differential mobile robot, which, in our case, includes pulleys and belts coupling, it is important to have a parametrization and fine tuning of the controllers.

The controllers are tuned by the internal model control method using MATLAB and LabVIEW to perform simulations and implementation of the controllers (Paiva et al., 2016). Through these practical tests, it will be made the tune refining. The goal of this work is to implement a PI controller to control the speed of each wheel of a differential robot, with empirical evaluations.

1.1 Differential Robot

Figure 1 shows the top view of the differential robot. This kind of model behaves in the same way our robot does and is mathematically equivalent, although there is a difference in the construction that will be further explained in Section 2. The robot head is indicated by the speed vector V. In the model shown, it is considered that both motor wheels are identical, with radius r, and independent actuators. The distance between the wheels, also called virtual axis, is 2L.

Figure 1 - Differential robot diagram showing the variables used to find the robot model.



Source: Adapted from (Tommasi et al., 2015).

The robot pose is defined by its position in the space and its orientation. Mathematically, it can be described as shown in Equation (1):

$$q = \begin{bmatrix} x \\ y \\ \theta \end{bmatrix}, \tag{1}$$

where θ is the robot angle orientation and (x, y) is the robot position in the Cartesian System. Point C is the robot center of mass, and it is used as a reference point (Tommasi et al., 2015). In the rear part of the robot, there is a castor wheel used to give complimentary support to the whole structure without interfering directly in the robot's movement.

The robot moves in a straight line when both wheels have the same speed, which means $v_R = v_L$, where v_R is the right wheel speed and v_L is the left wheel speed. If both speeds are positive, the robot moves forward; however, if both speeds are negative the robot moves backward. To allow the robot to make curves, the right and left wheels must have different speeds. To rotate right (clockwise), the left speed should be greater than the right speed ($v_L > v_R$). On the other hand, to rotate left (counterclockwise), it is necessary the opposite, which means that the right wheel speed should be greater than the left wheel speed ($v_R > v_L$) (Ayres et al., 2017).

Due to its nature and construction, this differential robot has nonholonomic constraints, which means it cannot move laterally. To reach lateral points, it should make a combination of maneuvers always moving forward or backward and making rotations. In short, it means that displacements directly over the x-axis are not allowed, since the wheels do not move only over this axis. The goal of this work is to present the strategy, highlighting its theoretical aspects and describing the steps needed to implement such control strategy. Furthermore, the potentials and limitations of this strategy are also analyzed, focusing on its application of autonomous navigation and programming.

2. Methodology

This work shows a physical structure of a robot that has the goal of autonomous navigation, based on conditions and application techniques that were gathered and compiled from a systematic revision of the literature (Romero et al., 2014). The methodology used is based on designing the robot structure and find the closest model approximation for the real world. All the research and experiments were made in the *Grupo da Automação Industrial* laboratory (GAIN), located in *Instituto Federal do Espirito Santo* (Serra – Brazil). The research is of qualitative nature, designing the robot, finding an accurate model of the robot for the real world, and programming a proper controller to allow the robot to be used as an educational mobile robotic platform.

The robot itself is composed of wheels, motors, encoders, and a metallic structure that houses all the embedded systems and devices. Additionally, there are batteries, voltage regulators, power drivers, and encoders, being the last one connected to a NI-MyRio device, from National Instruments, which controls each wheel. Following, it was built the algorithm that implements the PI controllers that actuate on each wheel separately, considering the robot model and its non-holonomic constraints.

One of the greatest challenges of mobile robotics is navigation. Thus, for each phase cited previously some speed evaluations were made in the differential robot to ensure the controller is acting properly.

2.1 Robot Structure

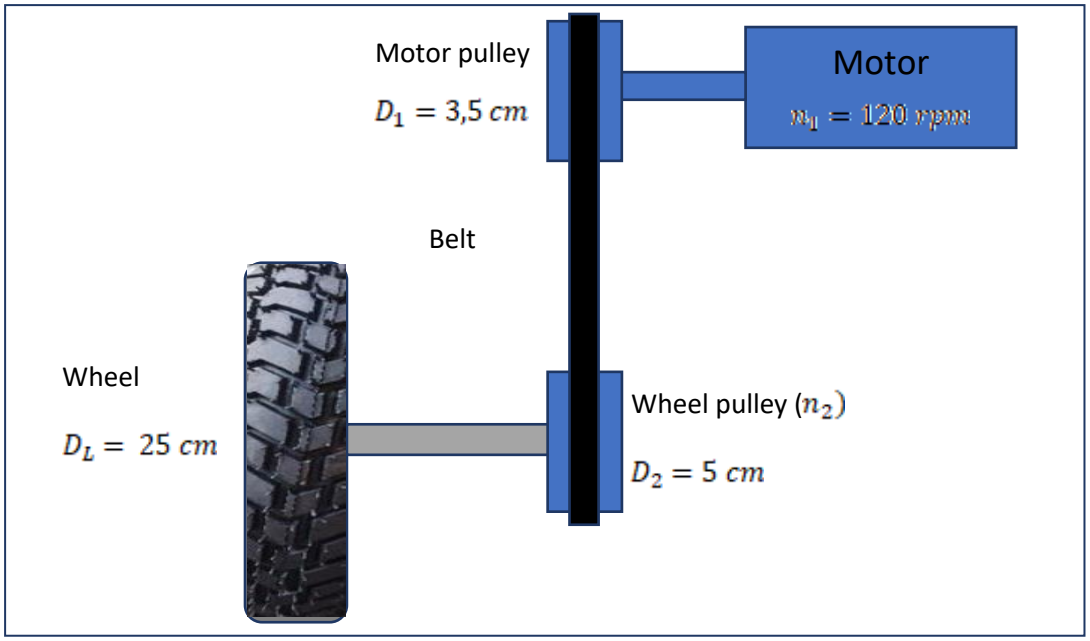
The designed robot has two independent parallel motor wheels in the front part and two wheels in the rear part that are connected with their respective in the front part. In other words, there are motor wheels in the front part and pulleys that connect the rear wheel to their respective motor front wheels. The two wheels are similar with a radius of 12.5 cm and the front wheels are separated by 37 cm (which is twice the distance between the centerline of the robot and each wheel). The mechanical structure of the wheels of one side is represented in Figure 1. There is a mirrored structure for the right part. The behavior and mathematical model of this robot are the same as the robot explained in Section 1.1, and, therefore, the same modeling used for that robot can be applied to this one.

Thus, due to its construction and mathematical model, our robot is differential with nonholonomic constraints, which means that some points in the Cartesian plane cannot be reached directly, such as those that require the robot to move laterally. To achieve some points on the plane the robot should combine rotational and translational movements by controlling the left and right wheel speeds.

In the design and conception of this robot, it should be able to move in both internal and external environments. Due to this feature, the robot should be robust, and its wheels should be neither large (to avoid problems in internal environments) nor small (to be able to move in external environments). Therefore, an intermediary size was chosen, which in the design of this robot corresponded to 25 cm in diameter. Another important specification of the robot is the maximum linear speed, which is 1.1 m/s. The robot design considered this important feature, and the robot has the sizes needed to allow the maximum speed of 1.1 m/s, as shown in Figure 2.

To build this robot it was taken as reference commercial robots, such as the one shown in (RobotShop, 2022) and the robot built by the *Industrial Automation Group* (GAIn) from the *Federal Institute of Espirito Santo*, in Serra, Brazil. Additionally, in the robot design and conception phase, it was considered the availability of motors in the market. Therefore, it was chosen the gear reducer MR 910-VE-120 RPM – 12Vcc with a torque of 32.8 kg.cm. However, due to the motor pulleys size relation, there is a speed reduction, which increases the torque in the wheels to 45.4 kg.cm (Cuadros et al., 2015).

Figure 2 - Left side mechanical diagram of the robot, showing the wheel, belt, pulleys, and motor connections with their respective features and dimensions.



Source: Own authorship.

Figure 2 shows how the proposed features were attended by designing the robot sizes and specifying the robot torque in the project phase. Equation (2) shows the relation of the motor and wheel pulley rotation speed and the diameter of those pulleys

$$\frac{n_1}{n_2} = \frac{D_2}{D_1}, \tag{2}$$

where n_1 is the rotational speed of the motor pulley, n_2 is the speed of the wheel pulley, D_1 is the motor pulley diameter, D_2 is the wheel pulley diameter, and D_L is the robot left wheel size (the right part of the robot has a mirrored structure with the same features and dimensions). Thus, using the values of $n_1 = 120\ rpm$, $D_1 = 3.5\ cm$ and $D_2 = 5\ cm$, it is possible to obtain the value of $n_2 = n_1 \frac{D_1}{D_2} = 84\ rpm\ = 1.4\ rps$, which, as aforementioned, is the rotational speed of the wheel. After obtaining such relations, it is possible to calculate the maximum wheel speed $v = n_2 \pi D_L = 110\ \text{cm/s}$, which corresponds to 1.1 m/s.

There are two pulleys on each side of the robot. One is the motor pulley that has a transmission relation with the other pulley connected to the encoder, which, in its turn, counts the number of pulses and, consequently, calculates the wheel speed. Each wheel has its encoder to measure the speed and the information of the four wheels are processed in the embedded system of NI-MyRio that uses an onboard Xilinx FPGA, thus allowing the encoder algorithm processing.

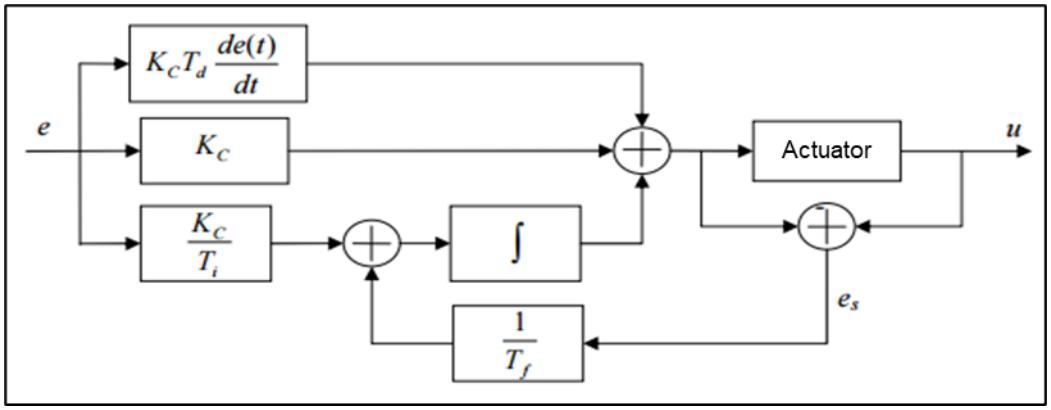
2.2 Control logic and programming

The way the PID controller was implemented followed the ideal configuration, as presented in Equation (3), where the constants are: T representing the period, K_c representing the gain, T_d representing the derivative time, T_i, the integrative time, and k the discretization variable.

$$u(k) = K_c\left[e(k) + \frac{T}{T_i}\sum_{i=1}^{k} e(i) + T_d\left(\frac{e(k) - e(k-1)}{T}\right)\right] \quad (3)$$

In a controller with an integral part, if the actuators reach their operational limits, the integral part keeps increasing (accumulating) the error without making any change in the actuator behavior, since it has already reached its maximum capacity. When the error changes its sign (from negative to positive or vice-versa), the integral part provokes a delay in the control action. Due to this feature, known as Wind-up, the process tends to become slow and oscillatory. To avoid this limitation, it was used a PID with anti-wind-up of type back-calculation, as shown in Figure 3, which has all variables in the time-domain (Atherton, 1995; Rundqwist, 1991; Vrancic & Hanus, 1996).

Figure 3 - Block diagram of the ideal PID with anti-wind-up controller used in our robot.



Source: Tommasi et al. (2015)

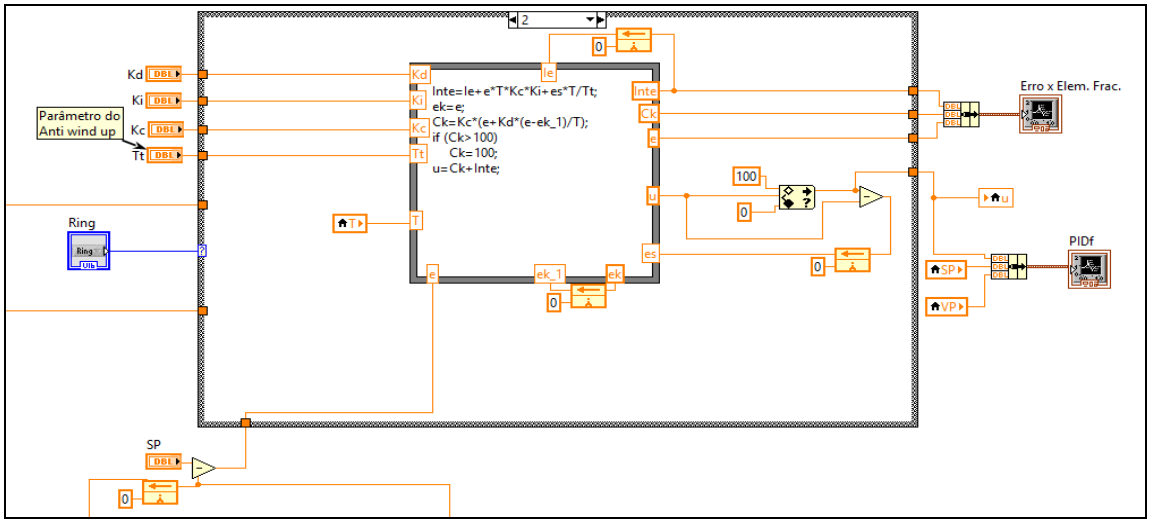
The time need for the input integrator to reach zero is determined by the gain $\frac{1}{T_f}$, where T_f can be viewed as a time constant that determines how fast the integer input will reach zero. Thus, in the first moment, the choice of small values for T_f can be an advantage. However, that choice should be made carefully, especially in systems with derivative action (Bernardes, 2017). Considering the PID with anti-wind-up configuration, Equation (3) was rewritten as shown in Equation (4).

$$u(k) = K_c\left[e(k) + T_d\left(\frac{e(k) - e(k-1)}{T}\right)\right] + T\sum_{i=1}^{k}\left(\frac{K_c}{T_i}e(i) + \frac{1}{T_f}e_s(i)\right) \quad (4)$$

Thus, considering the PID model and the robot requirements, it was made research about the available blocks in LabVIEW to make better planning of how the robot should be programmed.

Figure 4 shows the algorithm of the controllers programmed in LabVIEW considering the parameters of Equation (3) and the ideal PID with the anti-wind-up model.

Figure 4 - Ideal PID with anti-wind-up programmed in LabVIEW. Note the main code in the central block and the inputs and outputs that enter and exit the block.



Source: Own authorship.

The measurement of each robot wheel speed and the odometry is shown in the algorithm of

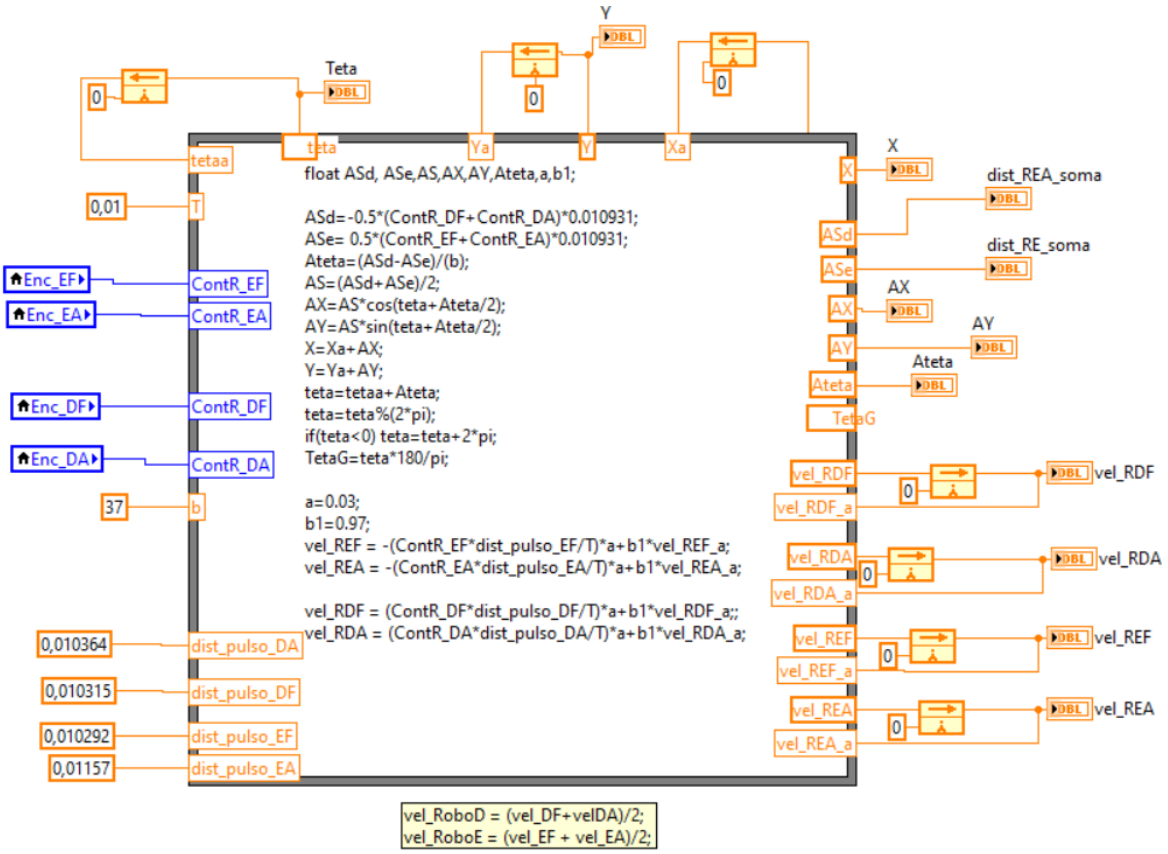
Figure 5. The speed calculation was made by counting the number of pulses in a certain amount of time; thus, knowing the distance related to a single encoder pulse, it is possible to compute the overall traveled distance. Once the information of the distance is known, it is only needed to divide the amount of time needed to travel that distance to find the speed (Bezerra, 2004), as shown in Equation (5).

$$v_r = \frac{Cont_r \cdot Dist\_Pulso}{T}, \quad (5)$$

where: $Cont_r$ is the number of pulses in the time interval T and $Dist_{Pulso}$ is the distance traveled by the wheel during the time needed to generate a single pulse. In

Figure 5, it is shown the algorithm that computes the speed for each wheel, and, at the same time, applies a first-order digital filter to avoid noise (Romero et al., 2014).

Figure 5 - Implementation of the robot speed measurement using LabVIEW. The mathematical equation of each wheel speed is represented in the yellow box below the main block.



Source: Own authorship.

3. Results and Discussion

In this section, it is shown how the methodology applied allows the system to be designed with an adequate controller, which demonstrated to be adequate for the proposed tasks of being able to be controlled moving in a straight path.

3.1 Robot Motor Wheels Modelling

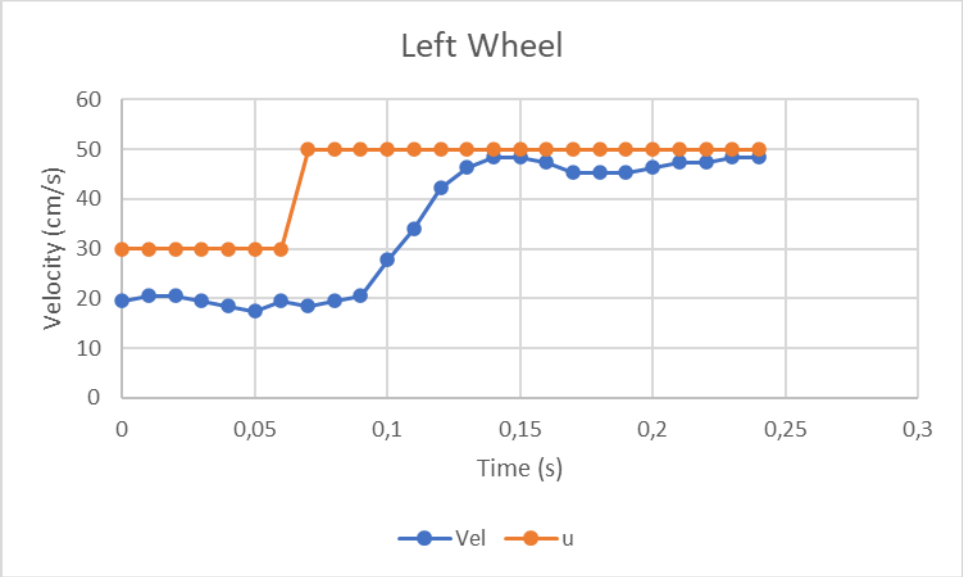
In this section, it is presented the models of each wheel that are used as references for the PID controllers' tuning. The input signal chosen to model the wheels was the step response. To obtain the individual model of each wheel the robot was suspended at a stable speed and the value of the initial control action *u* was further incremented applying a step signal of 20 units. This procedure was made for both right and left motor wheels.

3.2 The left wheel model

To obtain the left wheel model it was set a constant initial control signal *u* in 30 cm/s and, following, it was applied a step of 20 units (20 cm/s). The value of the signal was incremented from 30 cm/s to 50 cm/s, as shown in

Figure 6.

Figure 6 - Left wheel response to the step signal. The blue line represents the velocity of the robot starting from 20 cm/s and the orange line represents the step signal that changes the setpoint from 30 cm/s to 50 cm/s.



Source: Own authorship.

The features of the speed signal of the wheels were defined as a base to model the process and the tool used to find this model was the *ident toolbox* from MATLAB. To model the process response, it was used a first-order model with delay, represented by Equation (6).

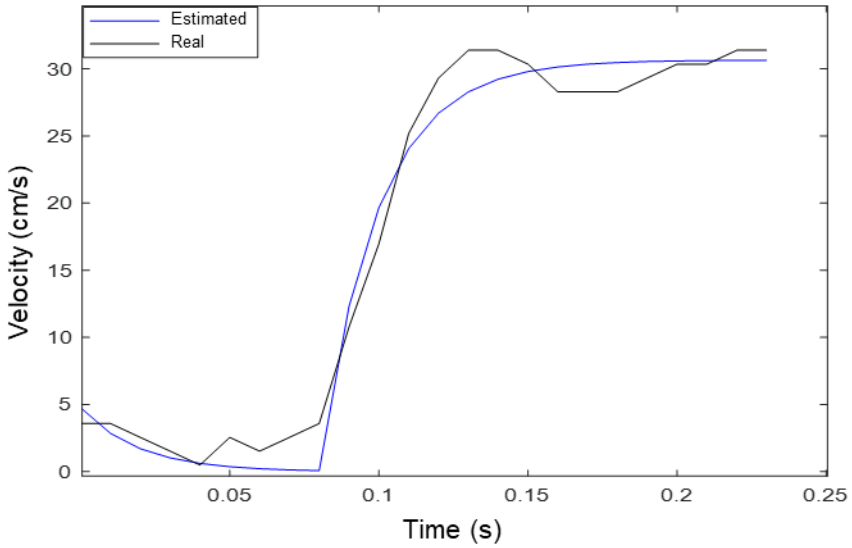
$$\left.\begin{matrix} K = 1.5054 \\ \tau = 0.0195 \\ \theta = 0.02 \end{matrix}\right\} G(s) = \frac{K}{\tau s + 1} \cdot e^{-\theta s}, \quad (6)$$

where K_p is the proportional gain, τ is the time the process response reaches 63% of the final value, and θ is the delay of the model to start reacting to the input signal change. To increase the amount of data to be used in the model identification, it was made an interpolation, which increased the data size by 10 times. Thus, the left wheel model found can be written as shown in Equation (7).

$$G_{RE}(S) = \frac{1,5054}{0,0195s + 1} \cdot e^{-0,02s}. \quad (7)$$

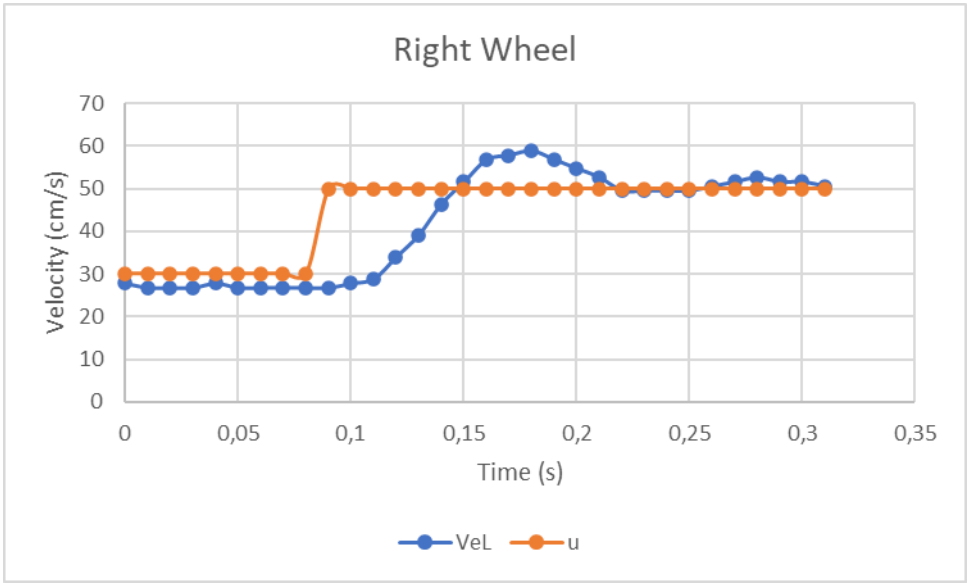
Figure 7 shows the expected and real results of the model found using the process response to a step input, while Figure 8 shows the variation of the input signal. This model was the one that obtained the better fit to the data collected and, additionally, it was the one that adjusted better to the real model behavior, thus helping to adjust better the controller.

Figure 7 - Step response and model of the left wheel. The "estimated" (blue) line shows the expected modeled output, while the "real" (black) shows the output given by the robot wheel.



Source: Own authorship.

Figure 8 - Applying the step signal to the right wheel. The blue line represents the wheel speed, while the orange line represents the step signal that changes the setpoint from 30 cm/s to 50 cm/s.



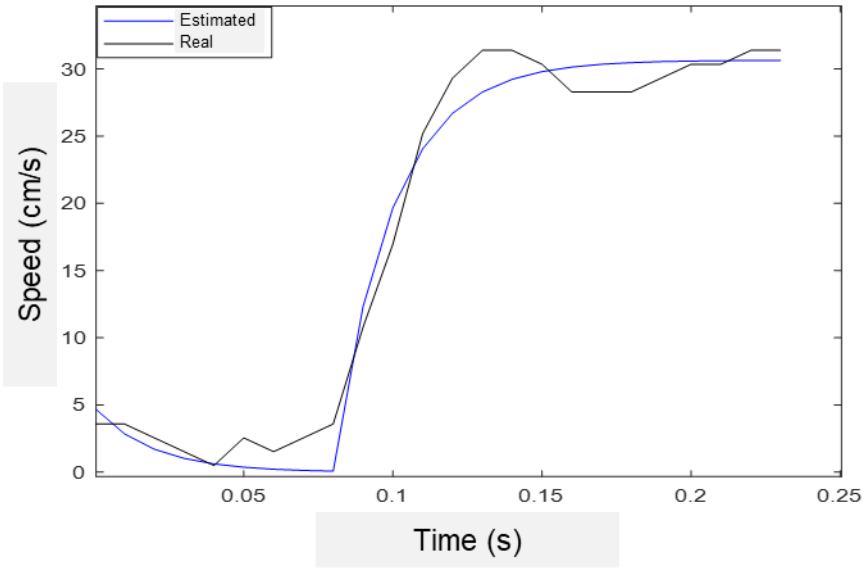
Source: Own authorship.

The characteristic equation of the right wheel behavior is given by Equation (8).

$$\left.\begin{matrix} K = 1.0493 \\ \tau = 0.0254 \\ \theta = 0.02 \end{matrix}\right\} G_{RDF}(s) = \frac{1,0493}{0,0254s + 1} . e^{-0,02s} \quad (8)$$

Figure 9 shows the expected and real results of the model found using the step response for the right wheel. Similar to the left wheel, the mathematical model for the right wheel has high correspondence to the collect data, being close to the real model.

Figure 9 - Step response to the right wheel. The "estimated" (blue) line shows the expected modeled output, while the "real" (black) shows the output given by the robot wheel.

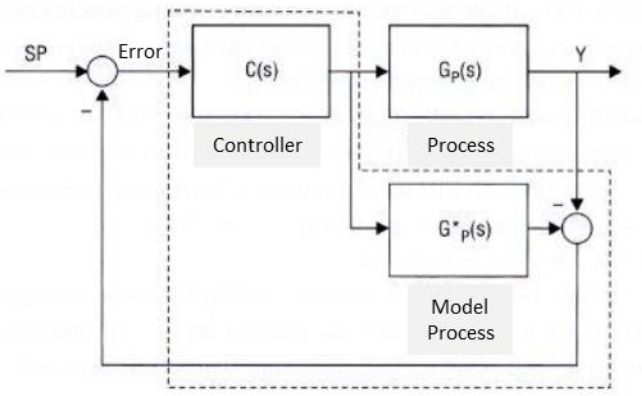


Source: Own authorship.

3.3 Tuning

The method chosen to tune the PID controllers was the Internal Model Control (IMC). In IMC, it is defined the performance criteria, represented by λ, to tune the controller considering the features and behaviors of the actuators (Campos, Mario César Massa; Teixeira, 2010). Therefore, a more robust tuning was applied using a high value of λ (about three times the time constant value of the process in open loop). Thus, the parameters of the PID controllers were determined as shown in the scheme of the IMC used to tune the controller presented in Figure 10.

Figure 10 – Internal Mode Control scheme used to tune the PID controller.



Source: Campos et al. (2010).

Then, it is possible to find the closed loop transfer function, shown in Equation (9), without the process model $G^*_p(s)$:

$$\frac{Y(s)}{SP(s)} = \frac{G_p(s) \cdot C(s)}{(1 + G_p(s) \cdot C(s))} \quad (9)$$

IMC is used to determine the best tunning for the controller $C(s)$ to the input step response of the system, represented by a first-order result transfer function in closed loop with a time constant λ, as shown in Equation (10).

$$\frac{Y(s)}{SP(s)} = \frac{1}{(\lambda S + 1)} \quad (10)$$

The factor λ, which is the only needed parameter to adjust, is the performance criteria of IMC and it defines how fast it is desired that the output follows the established setpoint. It was defined that λ equals 1 because it follows the time response in open loop. In Table 1 it is shown the calculations to find the PI or PID parameters using the process model and the λ value. The obtained parameters are shown in Table 2.

Table 1 - Tunning parameters using the Internal Model Control – IMC.

Controller	K_p	T_i	T_d	Performance Suggestion
PI	15.054	0.0195	0	1.17
PID	10.493	0.0254	0.02	1.524

Source: Rivera et al. (1986).

Table 2 - PI parameters using the IMC method.

PID	Model				Controller		
	K	τ	θ	λ	K_p	T_i	T_d
Front Left Wheel	15.054	0.0195	0.02	1.17	0.014	0.0295	0
Front Right Wheel	10.493	0.0254	0.02	1.524	0.019	0.0354	0

Source: Own authorship.

The dead time of the model was 0.02 s and, due to this reason, it was necessary to increase the constants K_p and T_i to higher values (especially K_p) and, thus, the system behaved with high variability. Therefore, it was made a manual adjustment and it was determined the parameters of the PI controllers, as shown in Table 3.

Table 3 - PI parameters using the manual adjustment.

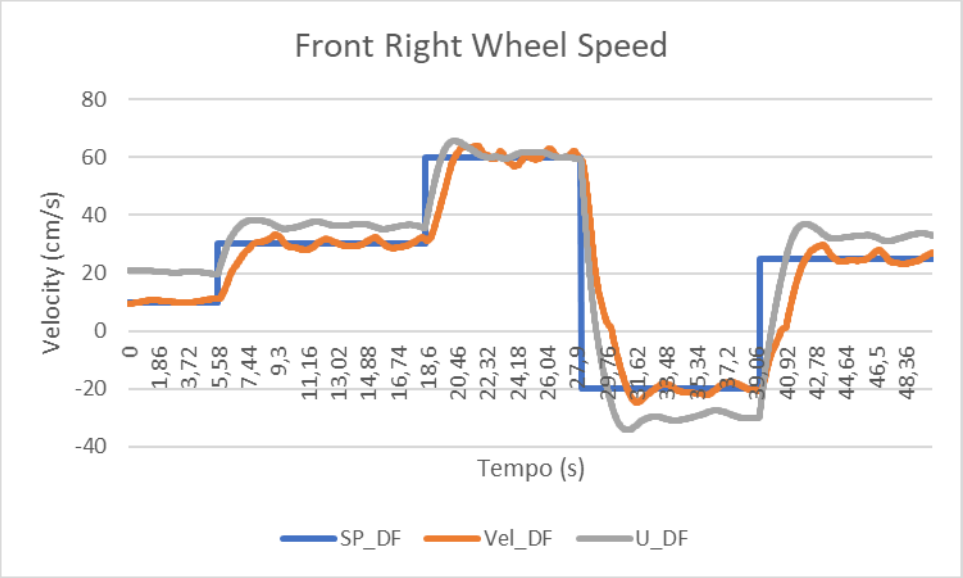
PID	K_p	T_i	T_d
Front Right	0,004	0,002	0
Rear Right	0,003	0,002	0

Source: Own authorship.

Once the PI parameters were set and all the algorithms coded and uploaded to the system, the robot was evaluated. The data acquired using LabVIEW was, then, exported to Excel to obtain graphics of both wheels in closed loop. Some setpoints were defined to make the robot move in a straight path in the GAIn laboratory, which has a smooth floor. The setpoints used were 13 cm/s, 30 cm/s, 60 cm/s, -20 cm/s and 25 cm/s in the trials performed in the laboratory that are displayed in Figure 11 and

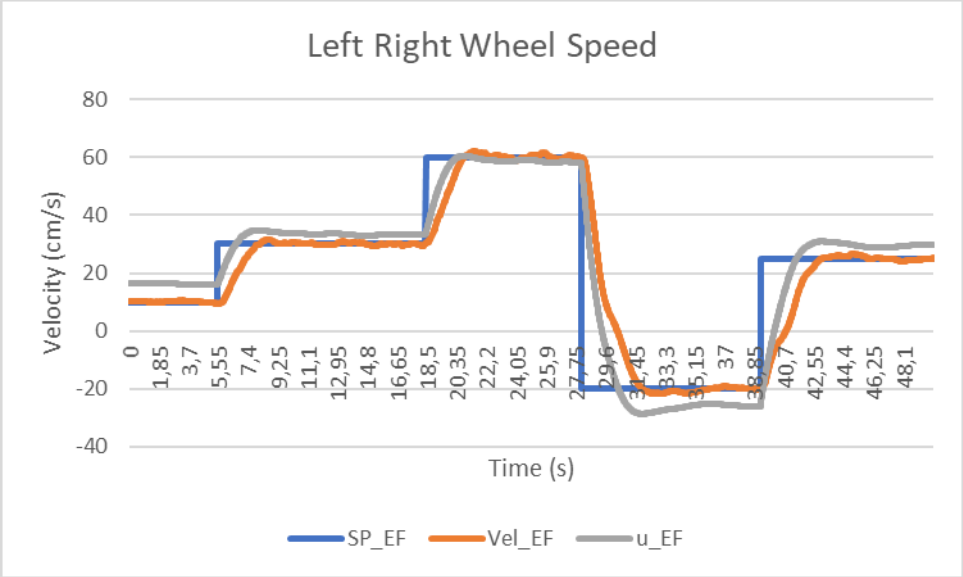
Figure 12. The results were considered satisfactory, with small errors and with a constant speed. The time response, however, had a short delay due to inertia. However, this did not affect the control system. Furthermore, it was placed a filter to minimize the noise effect.

Figure 11 - Closed loop response for the right wheel. The blue line is the setpoint, the gray line is the controller output signal, and the orange line is the wheel speed.



Source: Own authorship.

Figure 12 - Closed loop response for the left wheel. The blue line is the setpoint, the gray line is the controller output signal, and the orange line is the wheel speed.



Source: Own authorship.

4. Conclusion

In this work, it was developed and implemented PI controllers for each motor wheel of the robot and the results showed good performance of the PI controllers. The proposed controllers allowed the robot to reach the speeds they were projected to, and the first set of motor/reduction and power drivers worked properly. Readings from the wheels by the encoders were correct, showing that the FPGA functions in the embedded system NI-MyRio could make the correct measurement of the speed. Furthermore, the embedded system did not show memory issues, and the power drives were correctly triggered.

A PI controller was implemented to control the drive speed in the differential robot that was designed and built. Some improvements were made in the drives, electronic circuits, and embedded instruments of the differential mobile robot. Additionally, implementations of the PI controller to control the wheel speeds generated reliable results to control the robot. The experiments using the PI speed controllers tuned by IMC allowed maneuvers closer to the reference (setpoint). Thus, as a conclusion, we found that the IMC tuning allowed an optimal performance and adequate control of the mobile robot.

As future works, this platform will be used for obstacle avoidance, and to build maps of internal environments by doing SLAM (Simultaneous Localization and Mapping), thus increasing the possibilities of the platform for training in automatic control, instrumentation, and programming. To allow this, the robot will receive more sensors, such as laser sensors and cameras, and new codes will need to be implemented.

Acknowledgments

Authors would like to acknowledge the technical and financial support from *Grupo de Pesquisa da Automação Industrial* (GAIn) and from the *Instituto Federal de Educação, Ciência e Tecnologia do Espírito Santo* (IFES).

References

Åström, K. J. (2002). *Control System Design by Karl Johan Åström*. https://www.cds.caltech.edu/~murray/courses/cds101/fa02/caltech/astrom.html

Astrom, K. J., & Hägglund, T. (2006). Advanced PID control. In *ISA-The Instrumentation, Systems, and Automation Society*.

Atherton, D. P. (1995). An Analysis Package Comparing PID Anti-Windup Strategies. *IEEE Control Systems*, *15*(2), 34–40. https://doi.org/10.1109/37.375281

Ayres, L. M., Batista, L. G., da Silva, J. R., Motta, V. da R., Marques, V. M., & Cuadros, M. A. S. L. (2017). Desenvolvimento e Implementação de uma Arquitetura de Navegação Para um Robô Móvel Utilizando Comandos de Voz, Algoritmo A* e o Controlador Backstepping. *XIII Simpósio Brasileiro de Automação Inteligente - SBAI*.

Ben-Ari, M., & Mondada, F. (2018). *Elements of Robotics*. Springer International Publishing. https://doi.org/10.1007/978-3-319-62533-1

Bernardes, N. D. (2017). *Implementação do PID fracionário com filtro de Kalman em um robô móvel diferencial*. Federal Institute of Espirito Santo.

Bezerra, C. G. (2004). *Localização de um robô móvel usando odometria e marcos naturais*. Universidade Federal do Rio Grande do Norte.

Campos, Mario César Massa; Teixeira, H. C. G. (2010). *Controles típicos de equipaemntos e processos industriais*. Blucher, Ed. (2nd ed.).

Cuadros, D., A, M., Rogério, P., & Gamarra, D. (2015). Development of a mobile robotics platform for navigation tasks using image processing. In *Computer Science and Applications* (pp. 457–463). CRC Press. https://doi.org/10.1201/b18508-79

Faisal, M., Hedjar, R., Al Sulaiman, M., & Al-Mutib, K. (2013). Fuzzy logic navigation and obstacle avoidance by a mobile robot in an unknown dynamic environment. *International Journal of Advanced Robotic Systems*, *10*. https://doi.org/10.5772/54427

Manyika, J., Chui, M., & Bughin, J. (2013). Disruptive technologies: Advances that will transform life, business, and the global economy. *McKinsey Global …*, *May*, 163. http://www.mckinsey.com/insights/business_technology/disruptive_technologies%5Cnhttp://www.chrysalixevc.com/pdfs/mckinsey_may2013.pdf

Paiva, B., de Freitas, S., Medeiros, M. G., Ruella Da Silva, J., Maia De Almeida, G., Antonio De Souza, M., & Cuadros, L. (2016). *Utilização de exemplos criados no software LabVIEW ® implementados no starter kit 2.0 como ferramenta no ensino-aprendizagem da robótica*.

Rivera, D. E., Morari, M., & Skogestad, S. (1986). Internal Model Control, 4. *PID Control Design*, *1*, 252–265.

RobotShop. (2022). *Dr. Robot Jaguar 4x4 Mobile Platform - RobotShop*. https://www.robotshop.com/en/dr-robot-jaguar-4x4-mobile-platform.html

Romero, R. A. F., Silva Junior, E. P. e, Osório, F. S., & Wolf, D. F. (2014). *Robótica Móvel* (Vol. 1). LTC.

Rundqwist, L. (1991). Anti-reset windup for PID controllers. *IFAC Symposia Series - Proceedings of a Triennial World Congress*, *4*(8), 453–458. https://doi.org/10.1016/s1474-6670(17)51865-0

Sharma, R., Gaur, P., & Mittal, A. P. (2017). Optimum Design of Fractional-Order Hybrid Fuzzy Logic Controller for a Robotic Manipulator. *Arabian Journal for Science and Engineering*, *42*(2), 739–750. https://doi.org/10.1007/s13369-016-2306-0

Simoens, P., Dragone, M., & Saffiotti, A. (2018). The Internet of Robotic Things. *International Journal of Advanced Robotic Systems*, *15*(1), 172988141875942. https://doi.org/10.1177/1729881418759424

Souza, J. A. M. F. (2005). *Introdução aos robôs.*

Tommasi, E., Faria, H., Cuadros, M., Almeida, G., Resende, C., & Gamarra, D. (2015). Estudo Comparativo de Controladores de Seguimento de Trajetória para Robôs de Tração Diferencial: Fuzzy, Ganhos Fixos e Backstepping. *XII Simpósio Brasileiro de Automação Inteligente (SBAI)*, 1–6.

Vrancic, D., & Hanus, R. (1996). Anti-Windup, Bumpless, and Conditioned Transfer Techniques for PID Controllers. *IEEE Control Systems*, *16*(4), 48–57. https://doi.org/10.1109/37.526915

Y.Abdalla, T., & I. Hamzah, M. (2013). Trajectory Tracking Control for Mobile Robot using Wavelet Network. *International Journal of Computer Applications*, *74*(3), 32–37. https://doi.org/10.5120/12866-9700