Available online at www.sciencedirect.com

ScienceDirect

IFAC PapersOnline CONFERENCE PAPER ARCHIVE

IFAC PapersOnLine 53-2 (2020) 17518–17523

Low cost two-wheels self-balancing robot for control education powered by stepper motors

J.A. Borja*, I. Alvarado*, D. Muñoz de la Peña*

* Dpto. de Ingeniería de Sistemas y Automática, Universidad de Sevilla, Avda de los Descubrimientos s/n 41092, Sevilla

(e-mail: jose.b.industrial@gmail.com, ialvarado@us.es, dmunoz@us.es)

Abstract: This paper presents a low cost, two-wheels self-balancing robot for control education powered by stepper motors developed at the University of Seville. This design improves a previous model based on DC motors that has been used for the last five years in different courses in which students learn electronics, computer programming, modeling, control and signal processing by means of the construction and control of this robot. The new design improves the performance and reduces the total price of the device.

Keywords: Control education, computer programming, modeling, signal processing and project-based learning.

1. INTRODUCTION

Project based-learning methods using practical applications have a special interest in control education. The objective of these methods is that students come into direct contact with real problems. In this way, the motivation of the students is encouraged, since he they see how the subject they are learning can be applied. In addition, it makes students learn the subject day-to-day, leads to self-assessment, the teacher can continuously check and improve the educational method he imparts by making use of the progress of work, encourages research, etc.

However, these methods are often difficult to implement due to their high cost. This is due to the need to carry out experiments with real systems. One solution is to use simulated systems, but it is not always possible, because there are many features that may not appear in simulations, which in the real system may be critical.

The inverted pendulum has been used for control engineering at least sixty years (Roberge, 1960), and, remains at public interest. This system has been so popular due to have very attractive behavior and a simple structure, becoming used to transport people (Nguyen et al., 2004). Boubaker, O. (2012) presented a comprehensive historical analysis on the evolution of the inverted pendulum, emphasizing its academic use (with real or simulation systems). In this field, there has been particular focus to the price of the equipments (Manabe, 1995). The current prize of electronic components has allowed the construction of low cost platforms, increasing accessibility to the use of real systems for education.

This paper is a continuation of Gonzalez et al. (2017). In this work, the description of a low-cost experimental self-balancing robot based on Arduino was presented. This design has been used for over five years in courses in which students learn electronics, programming, modeling, control and signal processing through the construction and control of this robot. The resulting model was a multivariable unstable nonlinear system with no minimum phase zero.

In this work, some of the problems identified of this design, both from the operational and educational point of views, have been solved with significant hardware and software changes. In addition, the new design has a significant price reduction and is available in an online repository for any student who wishes to build the robot and carry out the proposed practical exercises. The hardware can be bought at an affordable price or manufactured at home. The repository also has programs ready for installation divided into different modules, so that the student can choose which modules to use, being able to focus on different subjects.

The main hardware modification that has been made is the replacement of DC motors with encoder by stepper motors, which has allowed solving all operational problems. To include stepper motors, there have been other modifications that will be indicated throughout the article.

Fig. 1. Self-balancing robot

10.1016/j.ifacol.2020.12.2660

A method to control acceleration of the steeper motors have been developed in order to be able to use these accelerations as the robot inputs. This method is implemented in two algorithms described below.

The new design has been developed and used by numerous last-year students in different degrees (Croche (2014), González (2016), Cortés (2019)). All the students that have built and used the robot are really satisfied with the teaching methodology and the challenges tackled.

This paper is organized as follows; first, the prototype hardware is presented followed by the reasons why the project was carried out and the strategies and algorithms implemented. Next the control system and some simulation and experimental results, comparing them to the previous version, are shown. Finally, the paper ends with some conclusions and future works.

2. BUILDING THE VEHICLE

The vehicle has been designed so the students can assemble the different parts, including all the connections between the different electronic devices. Figure 2 shows the connections between the different components.

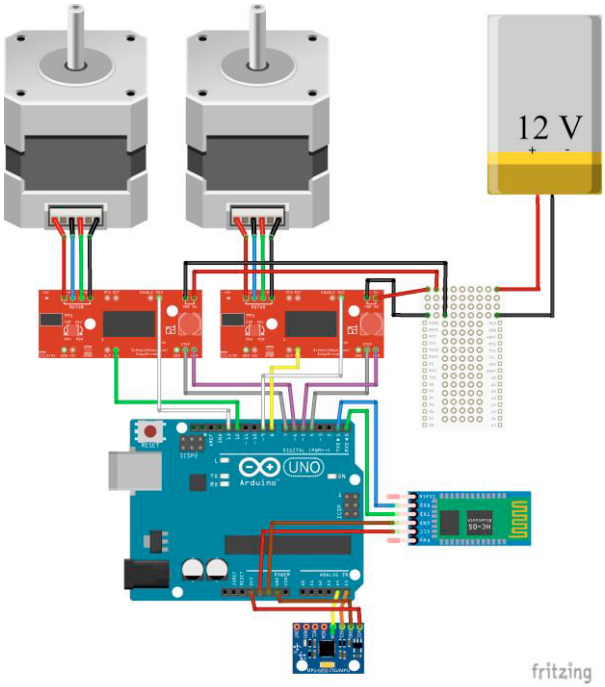

Fig. 2. Connections (Fritzing, 2016)

2.1 Inertial measurement unit (IMU) MPU6050

This is an electronic device that measures the tilt angle and the tilt angular speed using a combination of accelerometers and gyroscopes. The communication between this sensor and the microcontroller is done by I^2C.

2.2 Microcontroller Arduino UNO/NANO

The microcontroller reads the measurement of the tilt angle and the tilt angular speed from the IMU. Once it has all the variables, it controls the angular speed of the wheels by means of two PWM signals. It also performs all the required signal processing. In the previous design, it was necessary to use an Arduino MEGA, which has higher features and therefore higher price, due to the use of encoders, which are not necessary in this case

2.3 Microstepping motor driver A3967

The A3967 is a driver for bipolar stepper motors that performs the signal adaptation to control the motor. This device carries out the same task as motor controller cards for DC motors.

2.4 Motors Nema 17

The use of these stepper motors is the main difference with respect to the previous design based on DC motors. In the new version, bipolar stepper motors are being utilized. This motor is a brushless DC motor in which the complete turn is divided into several steps. Working in the correct operating range, they allow to maintain the position without using encoders. The speed of the motor can be controlled by means of a PWM signal y the direction with a digital signal using the microstepping motor driver A3967.

2.5 Bluetooth unit HC-06

To communicate online with the robot with an external device such as a mobile phone or a laptop computer, a Bluetooth unit was used. This Bluetooth unit enables the design of a remote controller for the robot.

2.6 Body

This component replaces the aluminium and wood chassis of the previous design. To make the body there are two possibilities: using a 3D printer or a wood laser cutter. The design for 3D printer was created in “STL” format. It is composed by 3 pieces: body and two wheels. They are assembled obtaining the full body. The design for laser cutter was created in “CDR” format. It is composed by different wooden plates (front, back, top, bottom, sides, etc.) that fit with each other obtaining the full body.

2.7 PCB

The different electronic components of the control system are interconnected by means of a PCB (easyEDA), which implies that the use of wires is almost removed. Using the body and the PCB a structure in which all the electronic components can be placed is obtained.

2.8 Cost

A cost reduction of approximately 50% has been achieved with respect to the price of the previous design (€ 110).

Devices	Cost
IMU MPU6050	2.5€
Arduino UNO/NANO	22€
Drivers A3967	2.5€ (x2)
Motor Nema 17	10€ (x2)
Bluetooth unit HC-06	3€
PCB	1.5€
3-D printed body	1€
TOTAL	55€

Table 1. Cost

3. REASON FOR THE CHANGES AND IMPROVEMENTS ACHIEVED

The main operational problems of the previous design were caused by the DC motors with gearbox in which the speed feedback was obtained from the measurement of double quadrature encoders. This scheme had the following disadvantages:

1. Clearances in the gearbox.

2. Dead zone of DC motors.

3. High noise in the speed and acceleration measurements obtained from the encoders.

In the design presented in this paper the DC motors have been replaced by stepper motors that solve all these issues.

These motors have high precision (1600 eighth step-by-turn) and enough torque, therefore it is not necessary to have a gearbox, so there are no clearances or dead zone.

Although the problem of dealing with dead zones and clearances is very interesting, we consider that, for introductory control courses, is better to avoid these issues and focus on linear control strategies such as PID. For advanced control courses in which clearances and dead zones compensation strategies are studied, the design based on DC motors can be used.

Furthermore, in the new robot, the speed and acceleration are known, since they depend on the pulse train sent to the driver. Therefore, no encoder is needed, and motor speed and position are known accurately.

However, the use of these stepper motors leads to new software problems. In the previous project the control action of the upper layer control was the rotational acceleration of the wheels. Two PID controllers were used to calculate the real acceleration, and the necessary voltage was applied to the motor. This structure changes dramatically with stepper motors. Both PID controllers are replaced by two algorithms, one that controls the speed, and another that modifies the speed reference so that a desired acceleration is applied.

4. ALGORITHM TO CONTROL SPEED

The problem of controlling speed and acceleration of stepper motors is challenging and has been studied before. *AccelStepper* is open access library for Arduino that allows controlling up to four stepper motors and supports acceleration and deceleration. However this library has not been used in this project for two reasons. First, it does not allow switching between the different micro-step configurations while it is running. So, the best resolution is not achieved. And second, the maximum speeds they allow are much lower than those that can be achieved by the motors. Thus, a new algorithm has been created, which is explained below.

To have control over the acceleration first we need to have control over the speed. In this design there are two stepper motors and their drivers. To move the motors it is necessary to generate a PWM with a frequency inversely proportional to the angular velocity. In general, the motors will have different speeds, in order to allow the rotation of the body, so two independent signals are needed, that is, a different train of pulses for each motor.

Arduino UNO has three timers, and the resolution of timer 1 is much higher than the others. So, it is better to control the motors with timer 1 and use timer 0 and timer 2 for other purposes.

This is achieved using interrupts A and B of timer 1, one for each motor. The algorithm decides the number of increments of the timer counter that corresponds to the period of the pulse train, considering that timer 1 has a resolution of 65536 (16 bits) and his prescale is set to divide-by-8 (32.77ms maximum period). The numbers of pulses are npulsA and npulsB, each corresponding to one of the stepper motors. The timer is programmed in normal mode, so it always counts to its maximum value. Figure 3 shows the pulse train strategy for interrupt A.

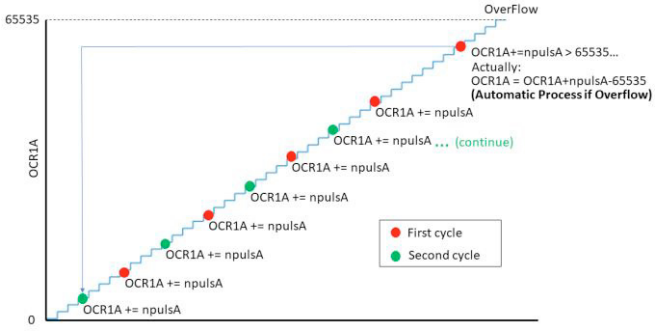

Fig. 3. Pulse train strategy using an interrupt and timer to OverFlow

The operation consists on self-increasing the value when the interruption goes off. When the counter, OCR1A/B, exceeds

the maximum (65535, 16bits), truncation is done automatically which implies that it is not necessary to make checks and resets, allowing maximum optimization.

This strategy allows us to control both stepper motors. Note that the code of these interruptions must be minimal, since they are executed much more frequently than the other processes.

The procedure described solves the problem of generating two different pulse trains for the movement of the motors. The only limitation is that the increase for the next interruption may not be greater than 65535, and that it the corresponding speed must be lower than the top speed allowed by the motors or drivers due to their physical features.

5. ALGORITHM TO IMPLEMENT ACCELERATION

Controlling directly the acceleration of the stepper motors is not possible, so, it is implemented a strategy in which the motor speed is updated periodically, generating a discrete acceleration profile as shown in figure 4.

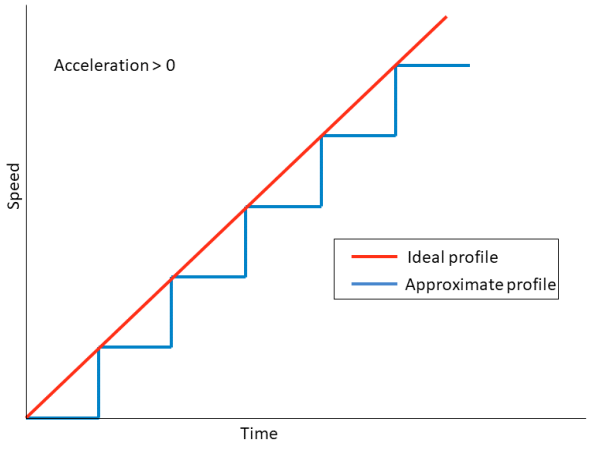

Fig. 4. Ideal speed profile versus approximate

When calculating the speed at each update, many issues must be considered and several operations must be carried out:

- Check if the speed is in the operating range.

- Check if it is positive or negative and act on the corresponding pin of the driver, DIR.

- Check in which ranges of micro steps can be generated to select between full step, half step, quarter step and eighth step to provide the highest accuracy.

Timer 2 is used update the speed values carrying out these operations. This timer has a resolution of 255 (8 bits). It is programmed in normal mode, with a prescale set to divide-by-to 256. The Overflow interrupt is generated every 4.08ms.

The first idea was to update the speed value each motor step. That is, a pulse is sent to the motor and it is calculated how long the next one must be sent. This implementation would be ideal; however, it was not possible to implement because each iteration consumes approximately 0.1ms, within interruptions that occur, on average, every 0.3ms. So, there would be interference.

If the sample time is smaller than the sample time of the calculation of the acceleration (four times faster), the method can be implemented and provides good results. This is the approach used in this design.

6. SYSTEM CONTROLLER

The system can be controlled using different strategies with different advantages and properties. First, students can use a PID controller whose input variable is the angle of the body and the output is necessary acceleration of the wheels to keep balance. It is the simplest control, but its results are not very good, since the speed of the wheels is not controlled, resulting in unwanted movements.

The next step would be to include another PID whose input is the angular speed of the wheels, and the output is the reference angle of the body for the PID from the previous paragraph, eliminating unwanted movements.

Although with these two PIDs the system control is acceptable, the result is drastically improved using an LQR, whose input variables are the angle and angular speed of the body with respect to the ground and the angular speed of the wheels, and the output variable is the necessary acceleration of the wheels to keep balance. In addition, an integral term is added to correct the difference between the center of gravity of the vehicle and the geometric one.

The resulting control law is:

$$u_{k+1} = -Kx_k + K_e e_k \equiv \ddot{\theta}_{k+1} = -[570\ 41.07\ 10]\begin{bmatrix}\phi_{c_k}\\ \dot{\phi}_{c_k}\\ \dot{\theta}_k\end{bmatrix} + [0.3]\sum_{i=0}^{k}\dot{\phi}_{e_i}$$

It is run on *Arduino*, with a sample time of 16 milliseconds.

To calculate the control law the continuous time model introduced in the paper of González et al (2017) has been used:

$$(2a + c\cdot\cos(\phi+\phi_0)\,\ddot{\theta} + (c\cdot\cos(\phi+\phi_0) + 2b)\cdot\ddot{\phi} - \\ - c\cdot\dot{\phi}^2\cdot\sin(\phi+\phi_0) - d\cdot\sin(\phi+\phi_0) = 0$$

$$a = \left(\frac{3}{2}m_r + \frac{1}{2}M\right)\cdot R^2$$
$$b = M\cdot L^2$$
$$c = R\cdot L\cdot M$$
$$d = M\cdot g\cdot L$$

The model for control is explained and developed in *Appendix A* of Gonzalez et al. (2017).

7. SIMULATION AND EXPERIMENTAL RESULTS. CHANGES IN THE SPEED REFERENCE

To verify the correct behavior of the system, a series of tests have been carried out to analyze the behavior of the vehicle. In addition, the control scheme was tested by students in simulation using the nonlinear continuous time model introduced in González et al (2017). Dimension and weight parameters have been updated for the new design. The identification of the wooden frame robot can be found in Cortés (2019).

With the LQR controller implemented, the vehicle can be moved at a desired speed. The following figures show the evolution of the inclination and angular speed with respect to the vertical, and angular speed and acceleration of the wheels, when two high speed reference changes are applied (in opposite directions), with a time margin of 5 seconds.

In Figure 5 it is shown the angular speed of the wheels.

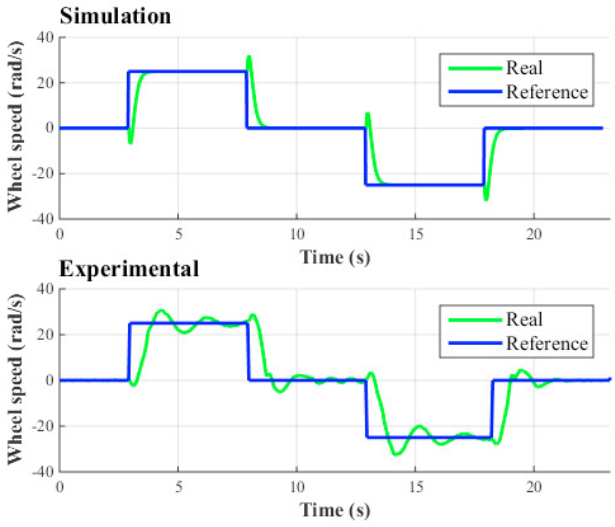

Fig. 5. Angular speed of the wheels

In Figure 6 it is shown the tilt angular speed trajectories when the different reference changes take place.

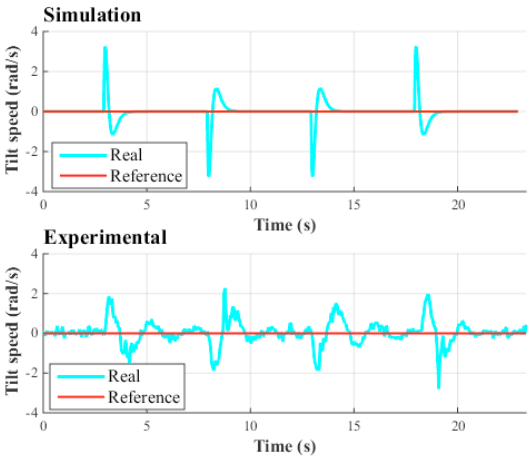

Fig. 6. Tilt angular speed

In Figure 7 it is shown the tilt angular trajectories when the different reference changes take place.

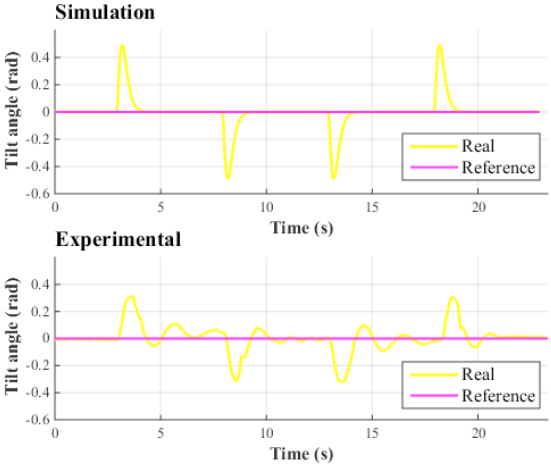

Fig. 7. Tilt angle

In Figure 8 it is shown the angular acceleration of the wheels.

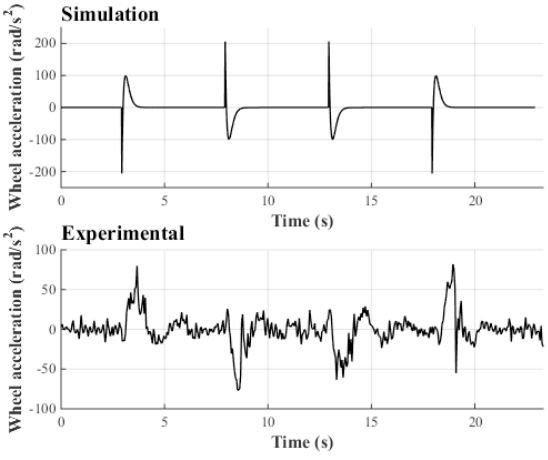

Fig. 8. Angular acceleration of the wheels

As it can be seen, when there is a step in the angular speed reference of the wheels, the acceleration increases or decreases as appropriate to achieve it. Overshoot usually occurs, but eventually the system stabilizes.

8. ANALYSIS AND CONCLUSIONS

As can be seen, in the figures corresponding to the acceleration and tilt speed of the simulations, peaks of great magnitude appear, as opposed to the graphs corresponding to the experimental results (see Figures 6 and 8). This is mainly because these peaks have a length lower than the sample time of data recording of the experimental system. However, except for these anomalies, all the magnitudes and profiles are very similar in both systems. So, the model simulation and the LQR controller have a correct operation.

Among the different graphs, wheel speeds stand out (see Figure 5). In the experimental system there are overshoots that do not appear in the simulations trajectories. This is due to differences between both systems. Probably, without ideal conditions, the time response to achieve the angular speed reference without overshoots would be too long, having to increase the controller gains. However, even with overshoots, the profile is very similar to the response provided by the model.

9. COMPARISON OF RESULTS BETWEEN PROJECTS

In order to analyze the improvements achieved, the same test has been performed on both systems. The test carries out is changing the wheel speed reference and waiting for the system to reach steady state. In Figure 9 it is shown the behavior system of the previous and new project in these tests.

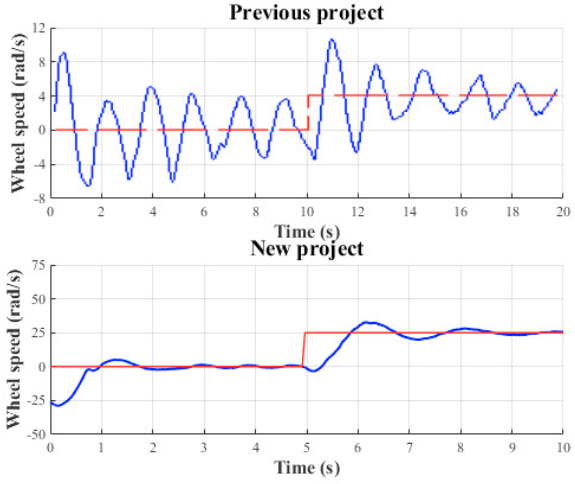

Fig. 9. Behavior of the previous project, wheel speed reference change

As can be seen, there is a drastic improvement. In the previous project overshoot is much greater than in the new project. In addition, in the improved system steady state is reached twice as fast, and oscillation amplitude around equilibrium are reduced.

10. FUTURE WORKS

A possible upgrade would be to control the system by means of an MPC in order to consider input and measurement constrains. Because of this kind of controllers are computing demanding, the Arduino will have to be replaced with a more powerful device.

11. REFERENCES

C. González, I. Alvarado, y D. Muñoz La Peña (2017). Low cost two-wheels self-balancing robot for control education. IFAC, PapersOnLine, 50(1), 9174-9179

Borja, J.A. (2018). Desarrollo de un robot autoequilibrado basado en Arduino con motores paso a paso (Diploma thesis). Universidad de Sevilla.

Croche, A.J. (2014). Vehículo autoequilibrado de tipo péndulo invertido de bajo coste (Diploma thesis). Universidad de Sevilla.

González, C. (2016). Mejora del software de un vehículo autoequilibrado de tipo péndulo invertido de bajo coste (Diploma thesis). Universidad de Sevilla.

Cortés, N. (2019). Diseño, fabricación, montaje, estudio dinámico, control y teleoperación de un vehículo tipo péndulo invertido sobre dos ruedas (Diploma thesis). Universidad de Sevilla.

Roberge, J. K. (1960). The Mechanical Seal (Bachelor's Thesis). Massachusetts Institute of Technology.

Boubaker, O. (2012, July). The inverted pendulum: A fundamental benchmark in control theory and robotics. In International conference on education and e-learning innovations (pp. 1-6). IEEE.

Nguyen, H. G., Morrell, J., Mullens, K. D., Burmeister, A. B., Miles, S., Farrington, N., ... & Gage, D. W. (2004, December). Segway robotic mobility platform. In Mobile Robots XVII (Vol. 5609, pp. 207-220). International Society for Optics and Photonics.

Manabe, S. (1995). A low-cost inverted pendulum system for control system education. In Advances in Control Education 1994 (pp. 21-24). Pergamon.

Fritzing (0.9.3b) [Software] (2016). Retrieved from https://fritzing.org/

EasyEDA [Software]. Retrieved from https://easyeda.com/