Available online at www.sciencedirect.com

ScienceDirect

IFAC Papers Online CONFERENCE PAPER ARCHIVE

IFAC PapersOnLine 50-1 (2017) 9174–9179

Low cost two-wheels self-balancing robot for control education

C. Gonzalez*, I. Alvarado*, D. Muñoz La Peña*
* Dpto. de Ingeniería de Sistemas y Automática, Universidad de Sevilla, Avda de los Descubrimientos s/n 41092, Sevilla
(e-mail: cecigg12@gmail.com, ialvarado@us.es, dmunoz@us.es)

Abstract: This paper presents an experimental, Arduino based, low cost self-balancing robot developed at the University of Seville for control education. The main idea is that the students can learn electronics, computer programming, modeling, control and signal processing by means of the construction and control of this robot. The resulting model is a multivariable unstable nonlinear system with non-minimum phase zero. Experimental results obtained by students of the University of Seville are included to demonstrate possibilities of the prototype.

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

1. INTRODUCTION

Teaching control is in general a challenging task because control systems are always included in complex systems that often are simplified so that the student focus on the study of the dynamics of the controlled system. A popular pedagogy to deal with this issue is project-based learning (PBL), Perrenet et al. (2000), in which the curriculum is delivered in a set of problems which provides the starting point for the learning process.

One of the issues that PBL faces in control is the need of physical experiments. One option is the use of emulation-based virtual laboratories, see Goodwin et al. (2011) and the references therein. However, physical experiments have a pedagogical value on their own because the student has to solve numerous problems not present in virtual settings.

Nowadays the current price of electronics components provides educators the opportunity to teach control by means of the construction of low cost platforms. For example, in Hau-Shiue et al. (2013) a two-wheels self balancing robot is presented, in Rashied et al. (2016) an example of a ball and beam is shown and in Lilienkamp et al. (2004) a magnetic levitator is built to teach control.

This paper presents the results of a project developed in the University of Seville with the objective to design a low cost two wheels self-balancing robot that can be built and controlled by students in a course, following a PBL approach learning electronics, computer programming, modeling, control and signal processing. Figure 1 shows the resulting robot.

A self-balancing robot is a robot that tries to balance on two wheels as if it were an inverted pendulum. The idea is to maintain the robot in balance by applying to the motors, that move the wheels, the voltage to get the appropriate speed.

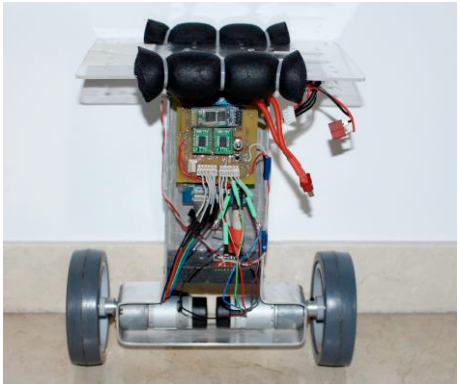

Fig. 1. Prototype design.

This paper will be organized as follows; first, the prototype hardware is presented. Then, a control scheme designed by students of the University of Seville is presented along some simulation and experimental results. The paper ends with some conclusions and future works.

2. BUILDING THE VEHICLE

The vehicle has been designed in a way such that the students can assemble the different parts, including all the connections between the different electronic devices. We present next the list of the hardware components used to build the prototype.

2.1 Inertial measurement unit (IMU) MPU6050

This is an electronic device 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.

10.1016/j.ifacol.2017.08.1729

2.2 Microcontroller Arduino Mega 2560

The microcontroller reads the measurement of the tilt angle and the tilt angular speed from the IMU and also reads the wheel encoders to obtain the angular speed of the wheels. Once it has all the variables, it controls the angular speed of the wheels by means of two PWM signals. It is also performs all the required signal processing.

2.3 Engine controllers TB6612FNG

The motors need more current than the Arduino can supply. Therefore it is necessary to amplify the PWM signal, calculated by the microcontroller. Two controllers are required to support the starting current of the motors.

2.4 Motors EMG-30

This vehicle is fitted with DC motors. The motor has a gearbox, which reduces its angular speed. The angular speed is reduced 30:1. The motor includes quadrature encoders. These quadrature encoders are two Hall sensors arranged in a 90º angle, which provide 360 pulses for each revolution of the wheel. The encoders provide a measurement of the angular speed and the direction. Each motor presents a dead zone and it also has a small clearance in the gearbox.

2.5 Bluetooth unit HC-06

To obtain live data, a Bluetooth unit was used. Furthermore this Bluetooth unit enables the design of a remote control for the robot, for example programming a smartphone APP.

2.6 Body

The body is made of aluminium and wood. All electronic devices are installed inside. With this structure, all the hardware elements are protected.

2.7 Connections

Figure 2 shows the connections between the different devices used in the robot. An integrated circuit was developed to simplify the connections.

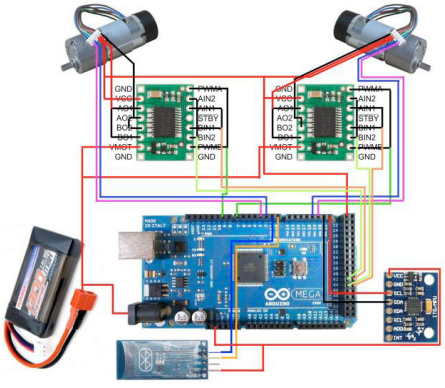

Fig. 2. Connections between the devices

For the prototype, this circuit was developed by students, however, depending on the course learning objectives it could be provided by the professor with the rest of the hardware.

2.8 Cost

Table 1 shows the cost of each device.

Devices	Cost
IMU MPU6050	2.48€
Arduino Mega 2560	17€
Controller TB6612FNG	3.82€ (x2)
Motor EMG-30	37.35€ (x2)
Bluetooth unit HC-06	9€

Table 1. Cost

3. CONTROLLING THE VEHICLE

There are multiple ways to design the control scheme for the robot built. We present next the configuration chosen for the prototype by the students based on two nested controllers, one to regulate the wheels acceleration and another to balance the robot. Figure 3 shows the different software elements of the control scheme including the signal processing blocks that obtain the measures from the sensors.

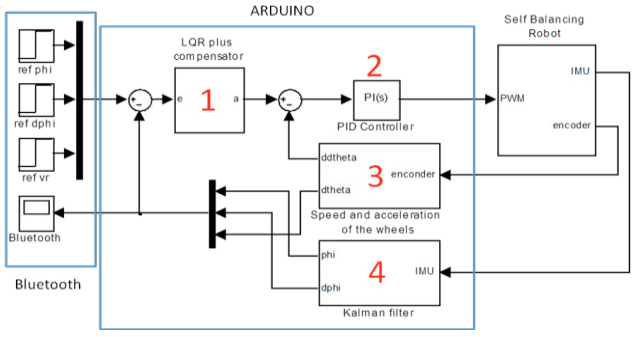

Fig. 3. Control block diagram

Four different programming tasks have to be accomplished by the student, shown in the figure 3 (numbers 1 to 4).

Block number 1 is the high level control. In this case it is a LQR controller that will control the tilt angle, tilt angular speed and angular speed of the wheels. The control action is the angular acceleration of the wheels that will be provided as a reference to two PIs that control this magnitude. The difference between the robot's gravity centre and its geometric centre makes the robot tends to move towards the gravity center. In order to cancel this perturbation an extra term has been added to the control law. How to compute this controller can be found in the appendix A.

Block number 2 is the control of the motors. In this case two PI controllers, one for each wheel, have been used to track the desired angular acceleration. The sample time chosen for these controllers is 10 milliseconds. The noise of the angle readings from the wheels increases as the sample time is reduced, so there is a trade-off between sampling speed and noise. In the experimental results this issue is discussed.

The PI control action is:

$$u_k = K_c\left(e_k + \frac{1}{T_i}\sum_{i=0}^{k} e_i\right), \qquad e_k = r_k - x_k \tag{38}$$

Where u_k is the PWM signals applied to the motors, x_k is the acceleration of the wheels and r_k is the desired angular acceleration provided by the LQR. The constant T_i is equal to the time constant obtained in the step response. Its value is 0.1. The constant K_c was obtained experimentally. Its value is 0.2.

Block number 3 is a filter to reduce the noise that affects the measurement of angular speed of the wheels. In this case a Butterworth filter was implemented.

Block number 4 is a filter to obtain a clean measure of the tilt angle and the tilt angular speed. In this case was used a Kalman filter provided by the manufacturer of the IMU.

4. SIMULATION RESULTS

As part of the curriculum, the control scheme was tested by the students in simulation using the nonlinear continuous time model presented in Appendix A. This is the same model used to obtain the linear discrete time model used to design the LQR gains.

The objective of the simulations is to demonstrate the correct operation of the LQR controller in different cases. In the proposed scenario, the robot starts the simulation away from the equilibrium point and the objective is to regulate the angle and the angular speed of the wheels to zero. After 10 seconds angular speed reference is changed 0.1. Then, at 20 seconds, once the controller stabilizes the angle, a load is suddenly added to the robot in a way such that the center of gravity is shifted to demonstrate that the controller is able to reject the error and maintain balance in a new steady state.

Figure 4 shows the evolution of robot's tilt angle. At time 10 seconds the angle deviates from the reference during a short time. This transient is due to the reference change in the wheel speed. At time 20 seconds, there is a sudden change in the tilt angle reference which rejects the effect of the load added to the robot.

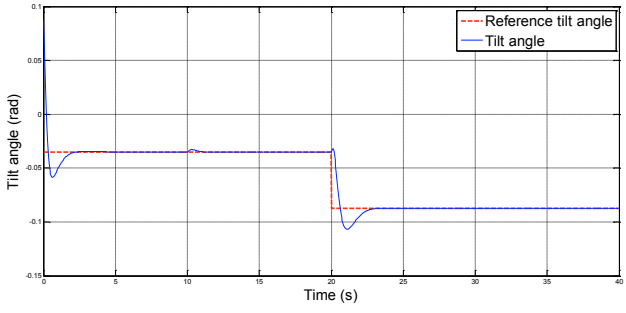

Fig. 4. Tilt angle of the simulation results

Figure 5 shows the tilt angular speed trajectories and the corresponding transients when the different reference changes take place.

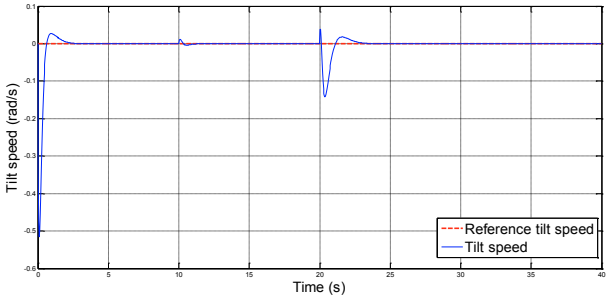

Fig. 5. Tilt angular speed of the simulation results

Finally Figure 6 shows the angular speed of the wheels.

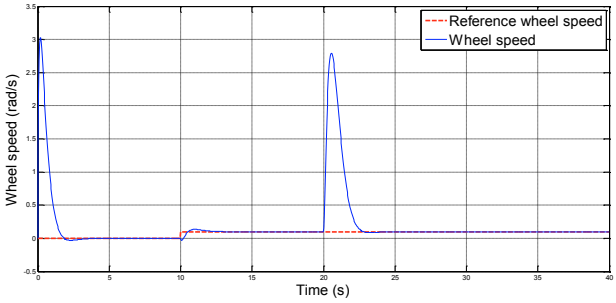

Fig. 6. Angular speed of the wheels of the simulation results

The simulations demonstrate that the LQR controller is well designed and that is able to control the nonlinear model.

5. EXPERIMENTAL RESULTS

In order to show the performance of the robot prototype 2 experiments have been realized by the students.

This first experiment shows a change of the angular speed of the wheels reference from -2 rad/s to 2 rad/s. Figures 7, 8 and 9 show the tilt angle, tilt angular speed and angular speed of the wheels trajectories for the this experiment respectively. The measurements obtained from the prototype sensors are subject to different sources of noises; however the controller is able to follow the references with an error smaller than 0.1 rad in the tilt angle and 0.6 rad/s in the angular speed of the wheels.

The main source of noise on this system stems from the measurement of the angular speed of the wheels. The way of estimate the angular speed is counting the number of pulses generated by the encoders in a fixed time (10ms), this implies that the value of the speed is discrete. Moreover the distribution of the pulses is not regular because the encoders are two Hall sensors arranged in 90º angle configuration. When the angular speed is high there is no problem with this way of measuring, but if the angular speed is low, then, the number of pulses is also small, leading to higher errors and to oscillations. This noise increases as the sampling time is reduced. The acceleration is obtained by numerical derivation, leading to an amplification of these errors.

Another source of noise is the IMU. This noise affects the measurement of the tilt angle and the tilt angular speed. In order to reduce this noise a Kalman filter provided by the manufacturer of the IMU is used. The amplitude of the filtered noise is small compared with the noise that affect to the angular speed of the wheels signal.

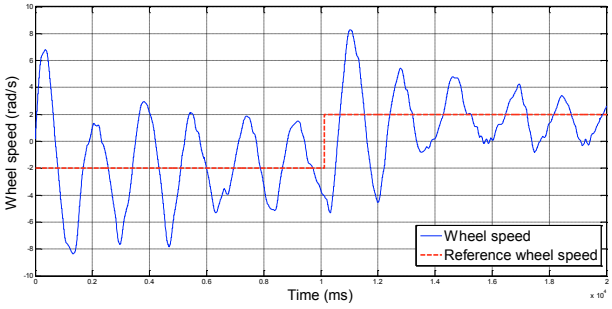

Fig. 7. Angular speed of the wheels versus time

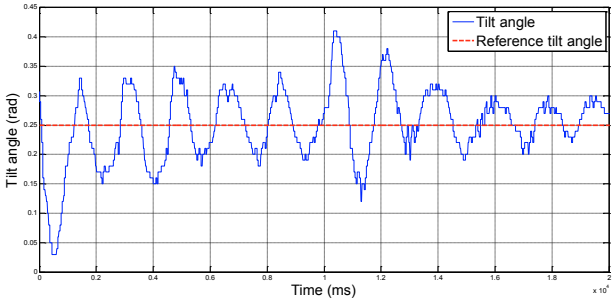

Fig. 9. Tilt angle versus time

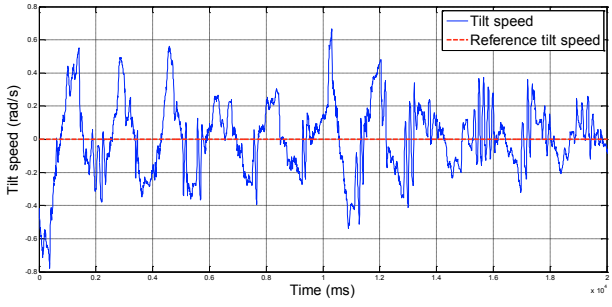

Fig. 9. Tilt angular speed versus time

The second experiment is the addition of a decentralized load. This load changes the robot's centre of gravity. Therefore tilt angle changes whereas the other parameters not.

Figures 10, 11 and 12 show the tilt angle, tilt angular speed and angular speed of the wheels trajectories for the this experiment respectively. In this experiment is also remarkable the noise due to the problem with encoders explained before.

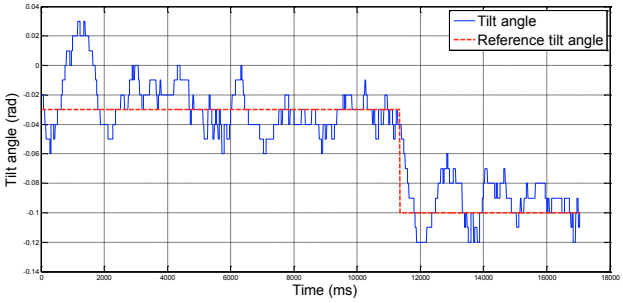

Fig. 10. Tilt angle versus time

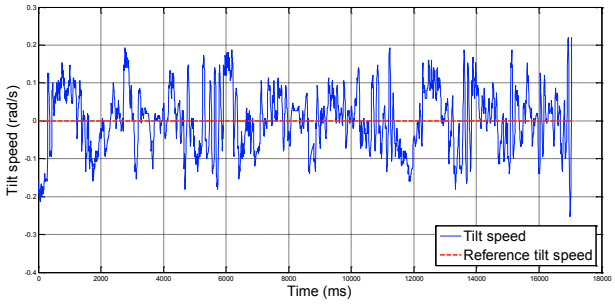

Fig. 11. Tilt angular speed versus time

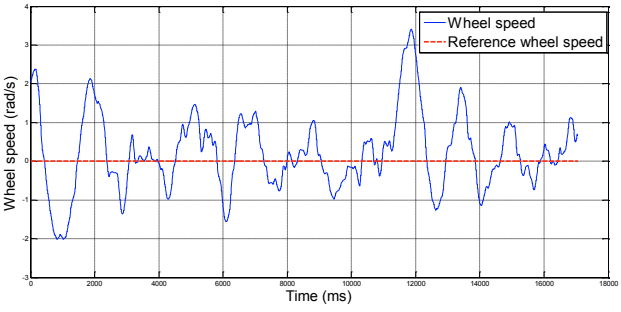

Fig. 12. Angular speed of the wheels versus time

6. CONCLUSIONS AND FUTURE WORKS

This paper presents an experimental, Arduino based, low cost self-balancing robot as an educational control system. This system has been the diploma thesis of two students (Croche, A. (2014) and Gonzalez, C. (2016)) but in future courses building and controlling the proposed robot will be included as part of the curriculum of control and robotics courses.

In addition, three new models will be designed; the first is a cheaper one (under 50€) as a laboratory system for the students of advanced control. The second one with a more powerful CPU for research and the third one is another low cost design using step motors instead of DC motors. The main objective of using these motors is to reduce the noise in the angular speed of the wheels measurements. For these three new prototypes, the body has been designed in a CAD program and will be built using a 3D printer. Figure 13 shows the new prototype body with the stepping motors.

Fig. 13. New self balancing robot model

7. REFERENCES

Croche, A. (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.

Goodwin, G.C., Medioli, A.M., Sher, W., Vlacic, L.B. and Welsh, J.S., 2011. Emulation-based virtual laboratories: a low-cost alternative to physical experiments in control engineering education. IEEE Transactions on Education, 54(1), pp.48-55.

Lilienkamp, K.A. and Lundberg, K. (2004). Low-cost magnetic levitation project kits for teaching feedback system design. Proceedings of the American Control Conference.

Hau-Shiue, J. and Kai-Yew, L. (2013). Design and control of a two-wheel self-balancing robot using the arduino microcontroller board. 10th IEEE International Conference on Control and Automation (ICCA), 634--639.

Rashied, Z., Hamees, M., Hassan, M.U., Hameed, S., and Khatri, N.A. (2016). Real time implementation of Robust PID controller for stabilization of Ball Balancing Beam. International Journal of Conceptions on Information Technology and Computing. 6--9.

Perrenet, J.C., Bouhuijs, P.A.J. and Smits, J.G.M.M., 2000. The suitability of problem-based learning for engineering education: theory and practice. Teaching in higher education, 5(3), pp.345-358.

Appendix A. Model for control

The LQR controller requires a linear model to be computed. Arduino requires the controller in discrete time. In order to obtain this a nonlinear model based on Lagrange equations is obtained, after that is linearized and discretized.

3.2 Non linear Model

To obtain the non-linear Model, it is going to use the Lagrange equations. The parameters are:

m_r, wheel's mass;

R, wheel's radius;

M, robot's mass without the wheels;

L, distance between the wheel shaft and the gravity centre;

g, gravitational acceleration.

ϕ_0, angle between the gravity centre and geometrical centre (unknown)

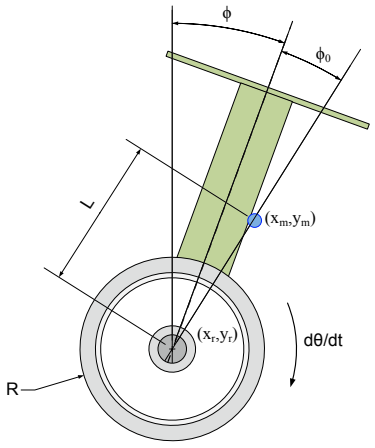

Fig. 14. System diagram with the variables

The generalized coordinates of the Lagrange equations are θ, angle between the wheel spin and the vertical, and ϕ, robot's tilt angle. For the model it is assumed that all the mass of the body is concentrated in a single point.

Lagrange equation:

$$L = T - V = T_{trans} + T_{rot} - V \tag{1}$$

Translational kinetic energy:

$$T_{trans} = \frac{1}{2} m_r (\dot{x}_r^2 + \dot{y}_r^2) \cdot 2 + \frac{1}{2} M (\dot{x}_m^2 + \dot{y}_m^2) \tag{2}$$

The wheel linear speed and the robot's gravity centre are:

$$x_r = R \cdot \theta; \quad \dot{x}_r = R \cdot \dot{\theta} \tag{3}$$

$$y_r = R; \quad \dot{y}_r = 0 \tag{4}$$

$$x_m = R \cdot \theta + L \cdot \sin(\phi + \phi_0) \tag{5}$$

$$\dot{x}_m = R \cdot \dot{\theta} + L \cdot \dot{\phi} \cdot \cos(\phi + \phi_0) \tag{6}$$

$$y_m = R + L \cdot \cos(\phi + \phi_0); \; \dot{y}_m = -L \cdot \dot{\phi} \cdot \sin(\phi + \phi_0) \tag{7}$$

then:

$$T_{trans} = \left(m_r + \frac{1}{2} M\right) \cdot R^2 \dot{\theta}^2 + \frac{1}{2} M L^2 \dot{\phi}^2 + MRL\dot{\theta}\dot{\phi} \cos(\phi + \phi_0) \tag{8}$$

Rotational kinetic energy:

$$T_{rot} = \frac{1}{2} m_r R^2 \dot{\theta}^2 + \frac{1}{2} M L^2 \dot{\phi}^2 \tag{9}$$

The potential energy is:

$$V = MgL \cos(\phi + \phi_0) \tag{10}$$

The result is:

$$L = \left(\left(m_r + \frac{1}{2}M\right)R^2 + \frac{1}{2}m_r R^2\right)\dot{\theta}^2 + ML^2\dot{\phi}^2 + MRL\dot{\theta}\dot{\phi}\cos(\phi+\phi_0) - MgL\cos(\phi+\phi_0) \quad (11)$$

It shows the Lagrange equation particularized in the generalized coordinates based on outboard motor torque:

$$\frac{d}{dt}\frac{\partial L}{\partial \dot{\theta}} - \frac{\partial L}{\partial \theta} = T \quad (12)$$

$$\frac{d}{dt}\frac{\partial L}{\partial \dot{\phi}} - \frac{\partial L}{\partial \phi} = -T \quad (13)$$

from (11), (12) and (13) the following equation is obtained:

$$(2a + c\cos\phi)\ddot{\theta} + (c\cos\phi + 2b)\ddot{\phi} - c\dot{\phi}\sin(\phi+\phi_0) - d\sin(\phi+\phi_0) = 0 \quad (14)$$

In which:

$$a = (m_r + M)R^2 \quad (15)$$

$$b = ML^2 \quad (16)$$

$$c = MRL \quad (17)$$

$$d = -MgL \quad (18)$$

It is remained to add the parameters values of the model.

Table 2. Parameters values

Parameter	Value
m_r	0.140 kg
M	0.82 kg
R	0.05 m
L	0.098 m
G	9.81 m/s^2

3.3 Linear Model for the control

In order to compute LQR controller is required a linear model.

It is considered small angles, ϕ<<1 then

$$\sin(\phi+\phi_0) \simeq (\phi+\phi_0); \quad \cos(\phi+\phi_0) \simeq 1 \quad (19)$$

The new simplify equation is

$$(2a + c)\ddot{\theta} + (c + 2b)\ddot{\phi} - c(\phi+\phi_0)\dot{\phi} - d(\phi+\phi_0) = 0 \quad (20)$$

The balance point is in:

$$\phi_{eq} = 0; \quad \ddot{\theta}_{eq} = 0 \quad (21)$$

Then the linearized equation is:

$$(2a + c)\ddot{\theta} + (c + 2b)\ddot{\phi} - d(\phi+\phi_0) = 0 \quad (22)$$

Assume that

$$e = 2a + c \quad (23)$$

$$f = c + 2b \quad (24)$$

Consequently, the system description in the state space will be:

$$\begin{bmatrix}\dot{\phi}\\ \ddot{\phi}\\ \ddot{\theta}\end{bmatrix} = \begin{bmatrix}0 & 1 & 0\\ -d/f & 0 & 0\\ 0 & 0 & 0\end{bmatrix}\begin{bmatrix}\phi\\ \dot{\phi}\\ \dot{\theta}\end{bmatrix} + \begin{bmatrix}0\\ -e/f\\ 1\end{bmatrix}\ddot{\theta} + \begin{bmatrix}0\\ -d/f\\ 0\end{bmatrix}\phi_0 \quad (25)$$

In a compact form:

$$\dot{x} = A_c x + B_c u + E_c w$$

Due to the Arduino works in discrete time, then a discrete time description is required. The sample time is 40ms. The following discrete time model is obtained using Matlab:

$$x_{k+1} = Ax_k + Bu_k + Ew_k$$

The LQR control computes the control action minimizing a cost function. The state vector is composed of robot's tilt angle, its speed and the wheel speed. The state weighting matrix Q and the control action weighting matrix R are:

$$Q = \begin{bmatrix}10 & 0 & 0\\ 0 & 1 & 0\\ 0 & 0 & 1\end{bmatrix} \qquad R = 0.1 \quad (27)$$

The resulting control law is:

$$u = -Kx = -[K_1 \quad K_2 \quad K_3]\begin{bmatrix}\phi\\ \dot{\phi}\\ \dot{\theta}\end{bmatrix} \quad (28)$$

Where K is:

$$K = [-332.8746 \quad -52.7835 \quad -3.1623] \quad (29)$$

Finally it was added a new term in the control due to the perturbation caused by the difference between the robot's gravity centre and its geometric centre. If they are not the same, the robot tends to move where more weighs.

In order to avoid the influence of this perturbation, the equation is formulated in incremental variables and discrete time.

$$\Delta x_{k+1} = A\Delta x_k + B\Delta u_k + E\Delta w_k \quad (30)$$

The perturbation is constant; therefore its variation is zero.

$$A\Delta w_k = 0 \quad (31)$$

Then the equation (30) is simplified:

$$\Delta x_{k+1} = A\Delta x_k + B\Delta u_k \quad (32)$$

The error in discrete time and incremental variables is:

$$\Delta e_{k+1} = \Delta r_{k+1} - \Delta x_{k+1} = -\Delta x_{k+1} \quad (33)$$

Consequently, the system is:

$$\begin{bmatrix}\Delta x_{k+1}\\ e_k\end{bmatrix} = \begin{bmatrix}A & 0\\ -I & I\end{bmatrix}\begin{bmatrix}\Delta x_k\\ e_{k-1}\end{bmatrix} + \begin{bmatrix}B\\ 0\end{bmatrix}\Delta u_k \quad (34)$$

Assuming that the control action is a gain:

$$\Delta u_k = K_z\begin{bmatrix}\Delta x_k\\ e_{k-1}\end{bmatrix} = K_x\Delta x_k + K_e e_{k-1} \quad (35)$$

Then it can be obtained that:

$$u_n = u_0 + K_x(x_n - x_0) + K_e\sum_{i=0}^{n-1} e_i \quad (36)$$

Where K_x is the LQR gain computed in (29). K_e is adjusted experimentally. K_e=10