Linear and angular position control of a custom built stepper motor driven self-balancing robot

Casian-George Iacob
Automation Department
Technical University of Cluj-Napoca
Cluj-Napoca, Romania

***Abstract*—The paper presents the full implementation of a self-balancing robot based on the mathematical model of an inverted pendulum. The self-balancing robot presented in this paper represents a real-life inverted pendulum built using stepper motors. A linear quadratic regulator (LQR) is tuned to obtain accurate control of angular and linear position of the robot. The control strategy is successfully validated on both simulation and real-life experiments targeting reference tracking and disturbance rejection.**

***Index Terms*—self-balancing robot, inverted pendulum, LQR control, stepper motors**

I. Introduction

Maintaining an inverted pendulum mounted on a cart in upright position by accelerating the cart back and forth represents a classic control theory problem. A self-balancing robot (SBR) is an adapted version of an inverted pendulum. The main difference is that an SBR consists of only one moving part and it can be controlled by the motors mounted at its base.

The robot is described by a single input multiple output (SIMO), unstable system with nonlinear dynamics which makes it hard to control. This is a classic control theory problem that has received numerous solutions during the years [1]–[9]. In most of these solutions the robot is actuated by geared DC motors.

The paper is focused on building and controlling an SBR using stepper motors and provide that this approach is also valid in the field of inverted pendulum control applications. A stepper motor has two main advantages over DC motors. The first one is that no additional gears are needed due to the high output torque it can deliver. The second one is that stepper motors rotate in discrete steps which yields high position control accuracy and allows open loop control.

A full state feedback, linear quadratic regulator (LQR) controller is approached since it is fully customisable on the needs of the process, making it an ideal control choice for a self balancing robot as shown in [8], [10]–[12].

The mathematical modeling of an inverted pendulum and the control theory behind an LQR controller are presented in the theoretical aspects. The adaptation of the SBR to the mathematical model of the inverted pendulum is next described. The main components used for building the robot are also presented. In the next section, an LQR controller is tuned based on the identified parameters of the real life platform. Last but not least an experiment is performed and the results are compared to numerical simulation data.

II. Theoretical aspects

A. Mathematical modeling

The free body diagram of an inverted pendulum mounted on a cart can be seen in Fig. 1. The two-degrees of freedom cart has a translational movement along the X axis, while the pendulum rotates around the Y axis, known as pitch in the specialized literature. The meaning of the parameters introduced by the figure will be explained further, as well as their influence on the physical system.

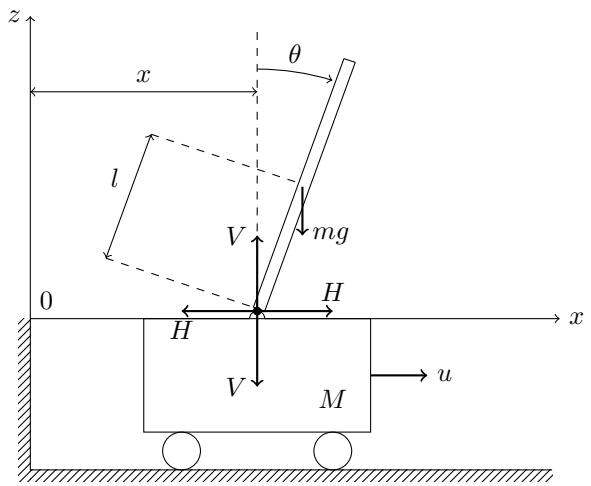


Fig. 1: Free body diagram of an inverted pendulum

The coordinates of the center of gravity (COG) of the pendulum are defined as

$$x_G = x + l\sin\theta$$
$$y_G = 0$$
$$z_G = l\cos\theta$$

where x is the linear position of the cart, l is the distance from the COG of the pendulum to the end connected to the cart and θ is the angular position of the pendulum from the vertical line.

By applying Newton's second law of dynamics for both linear and rotational movement, the following equations are obtained

$$I\ddot{\theta} = Vl\sin\theta - Hl\cos\theta \quad \text{(II-A.1)}$$

$$M\ddot{x} = u - H \quad \text{(II-A.2)}$$

where u denotes the control effort as a force, I is the mass moment of inertia of the pendulum about its COG, M is the mass of the cart. H describes the horizontal motion of the pendulum

$$H = m\frac{d^2}{dt^2}(x + l\sin\theta) \quad \text{(II-A.3)}$$

where m is the mass of the pendulum. V describes the vertical movement of the pendulum

$$V = m\frac{d^2}{dt^2}(l\cos\theta) + mg \quad \text{(II-A.4)}$$

The goal of the controller is to keep θ at the zero position, keeping the pendulum perpendicular to the ground, pointing upwards. Hence in a small region around θ = 0, the model can be linearized considering $\sin\theta \approx \theta$, $\cos\theta \approx 1$ and $\theta\dot{\theta}^2 \approx 0$ which yields

$$I\ddot{\theta} = Vl\theta - Hl \quad \text{(II-A.5)}$$

$$H = m(\ddot{x} + l\ddot{\theta}) \quad \text{(II-A.6)}$$

$$V = mg \quad \text{(II-A.7)}$$

From (II-A.2) and (II-A.6), we obtain

$$(M+m)\ddot{x} + ml\ddot{\theta} = u \quad \text{(II-A.8)}$$

Furthermore, combining (II-A.5) with (II-A.6) and (II-A.7) leads to

$$(I + ml^2)\ddot{\theta} + ml\ddot{x} = mgl\theta \quad \text{(II-A.9)}$$

Rewriting (II-A.8) and (II-A.9), the following mathematical model is obtained

$$\begin{aligned} \ddot{\theta} &= A_1\theta - B_1 u \\ \ddot{x} &= -A_2\theta + B_2 u \end{aligned} \quad \text{(II-A.10)}$$

where

$$A_1 = \frac{(M+m)mgl}{(M+m)I + Mml^2}$$

$$B_1 = \frac{ml}{(M+m)I + Mml^2}$$

$$A_2 = \frac{m^2l^2g}{(M+m)I + Mml^2}$$

$$B_2 = \frac{I + ml^2}{(M+m)I + Mml^2}$$

By denoting the states of the system as

$$\begin{bmatrix} \theta \\ \dot{\theta} \\ x \\ \dot{x} \end{bmatrix} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} \quad \text{(II-A.11)}$$

and considering the outputs of the system

$$\begin{bmatrix} \theta \\ x \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} \quad \text{(II-A.12)}$$

the following state-space representation is obtained

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \\ \dot{x}_4 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ A_1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ -A_2 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} + \begin{bmatrix} 0 \\ -B_1 \\ 0 \\ B_2 \end{bmatrix} u \quad \text{(II-A.13)}$$

Note that the outputs are

$$\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} \quad \text{(II-A.14)}$$

B. Full State Feedback

The working principle of a full state feedback controller consists of measuring all states of the system, multiplying them by a gain and subtracting them from a reference. In case of a non-zero reference, it must be scaled by a gain to obtain zero steady state errors. Another option for eliminating steady state errors is replacing the measured states by the current state errors. Hence the control law is

$$u = -\mathbf{K}\left(\mathbb{X} - \mathbf{Ref}\right) \quad \text{(II-B.1)}$$

where $\mathbb{X}$ is the state vector of the system, $\mathbf{Ref}$ is a vector containing the set point for each state, u is the control effort and $\mathbf{K}$ is a gain vector. The tuning of a full state feedback controller consists of computing the $\mathbf{K}$ that ensures the desired performances.

C. Linear Quadratic Regulator

An LQR controller is a form of optimal control. A cost function is set by the following equation

$$J = \int_0^\infty \left(\mathbb{X}^T\mathbf{Q}\mathbb{X} + u^T\mathbf{R}u\right)dt \quad \text{(II-C.1)}$$

where $\mathbf{Q}$ is a positive semi definite matrix of size $n \times n$, n being equal to the number of states and $\mathbf{R}$ is a positive definite matrix of size $m \times m$, m being the number of control signal inputs.

The selection of $\mathbf{Q}$ and $\mathbf{R}$ is done considering the desired performances of the closed loop system. In order for these to be positive semi definite and positive definite, respectively, they are defined as diagonal matrices with positive entries.

$$\mathbf{Q} = \begin{bmatrix} q_{11} & 0 & \dots & 0 \\ 0 & q_{22} & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & q_{nn} \end{bmatrix} \quad \mathbf{R} = \begin{bmatrix} r_{11} & 0 & \dots & 0 \\ 0 & r_{22} & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & r_{mm} \end{bmatrix}$$

The values of q_{ii} and r_{jj} are selected intuitively, and then readjusted according to the response of the system. A large value for q_{ii} is equivalent with a fast regulation for the state x_i yielding an aggressive control. Large values for r_{jj} are selected to avoid saturation of the control effort u_j.

After selecting $\mathbf{Q}$ and $\mathbf{R}$, the goal of the optimization problem is to get a minimum value for the cost function J. Considering the state-space representation of a system to be

$$\dot{\mathbb{X}} = \mathbf{A}\mathbb{X} + \mathbf{B}u$$

the solution of the previously defined optimization problem is given by a full state feedback controller with the gains defined as

$$\mathbf{K} = \mathbf{R}^{-1}\mathbf{B}^T\mathbf{S} \tag{II-C.2}$$

where $\mathbf{S}$ is the solution to the Algebraic Ricatti Equation [13]

$$\mathbf{A}^T\mathbf{S} + \mathbf{S}\mathbf{A} - \mathbf{S}\mathbf{B}\mathbf{R}^{-1}\mathbf{B}^T\mathbf{S} + \mathbf{Q} = 0 \tag{II-C.3}$$

The equation has multiple solutions but only one ensures closed loop stability. This can be found by computing the closed loop $\mathbf{A}$ matrix, denoted by $\mathbf{A}_{cl}$ which can be defined as

$$\mathbf{A}_{cl} = \mathbf{A} - \mathbf{B}\mathbf{K} \tag{II-C.4}$$

The solution of the equation is found in order to fulfill the condition that all the eigenvalues of matrix $\mathbf{A_{cl}}$ have a negative real part.

III. Real Life Platform

A. Description

A picture with the custom built SBR is presented in Fig. 2. The parameters of the mathematical model derived in subsection II-A need to be adapted such that the model describes the dynamics of the custom built SBR. Hence the following assumptions are made

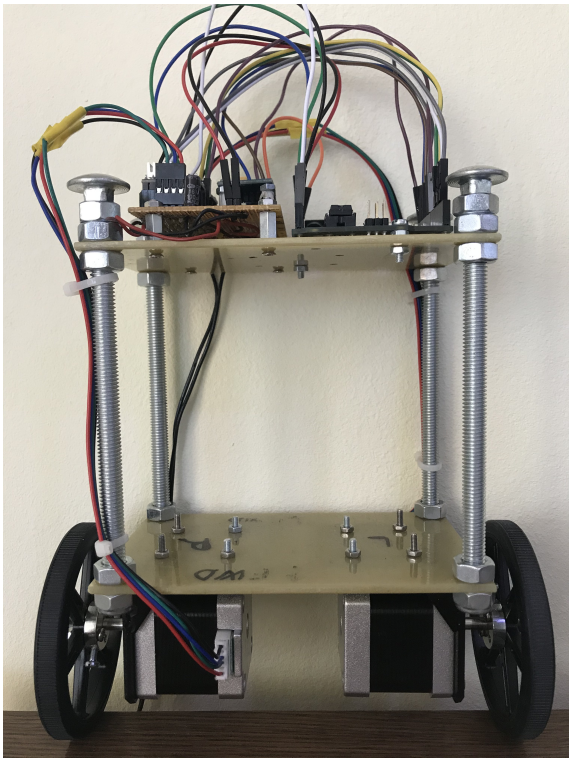


Fig. 2: Front view of the custom built SBR

- the cart is represented by the motor shafts and wheels
- the pendulum is represented by the robot body (motor stators, motor brackets, threaded rods, plastic sheets, electronics)
- the pendulum length is measured from the axis of rotation (the robot rotates about the motor shafts) to the plane in which the COG of the robot body lies.

It is assumed that there is enough friction between the wheels of the robot and the contact surface and that the mass of the robot is equally distributed on the horizontal planes. Furthermore, air friction is neglected. Based on the mathematical model presented in subsection II-A and the previously made assumptions, the following physical parameters presented in TABLE I were identified.

TABLE I: Robot Parameters

Parameter	Symbol	Value
Mass of the robot body	m	1.177 kg
Mass of the wheels and motor shafts	M	0.192 kg
Distance from rotation axis to COG	l	0.05 m
Mass moment of inertia of the robot body	I	0.0029 kgm^2
Gravitational acceleration	g	9.81 $\frac{\text{m}}{\text{s}^2}$
Wheel radius	r	0.04 m

An additional parameters is introduced, r, to denote the wheel radius. The two wheels chosen for the construction are identical.

B. Microcontroller

The control algorithm is implemented on an ATmega328p 8-bit microcontroller embedded to an Arduino UNO board. The microcontroller runs at a clock frequency of 16 MHz and has various features. The most relevant ones for this project are the I^{2}C interface, 8-bit timer/counter with programmable prescaler and compare mode and 23 input/output lines.

C. Actuators

Two bipolar stepper motors of size NEMA 17 are used for actuating the SBR. The motors can easily be driven using a dedicated driver such as A4988 or DRV8825. The drivers allow the motors to be pulse driven by sending impulses on a dedicated step pin from the breakout board to which the driver integrated circuit is embedded. At each rising edge of an impulse sent on the step pin, the motor shaft makes one full step equivalent with rotating 1.8°. This results in 200 steps needed to perform a full revolution. The pulse frequency gives the rotational speed of the motor shaft measured in steps per second. For a smoother rotation the drivers also allow microstepping. DRV8825 has been chosen due to its high microstepping resolution of 1/32 (i.e. $0.05625°/\text{step}$) proving an ideal choice for the proposed implementation.
The motors can reach a maximum rotational speed of 300 rotations per minute. This limit is high enough for the implementation of the self balancing robot and does not represent a risk of saturation.
Acceleration is torque dependent and torque is current dependent. Stepper motors normally deliver an almost constant torque at rotational speeds smaller than 300 rotations per minute, but only if enough current is supplied. This can be adjusted by the potentiometer on the DRV8825 driver.

The adjustment is done by setting the voltage drop across the potentiometer and ground, v_{ref}, based on the following equation

$$v_{ref} = \frac{\text{rated motor current}}{2}$$

The two identical motors used to engage each wheel in the SBR experimental setup have a rated current of 1.8 Amperes.

D. Sensors

The GY-521 breakout board featuring an MPU-6050 inertial measurement unit is used for measuring and estimating θ and $\dot{\theta}$. The MPU-6050 encapsulates a 3-axis accelerometer and a 3-axis gyroscope. The gyroscope data is used to measure $\dot{\theta}$, whereas the value of θ is estimated using accelerometer measurements and by integrating $\dot{\theta}$. The microcontroller accesses the data registers of the MPU-6050 via the I^2C bus.
Additionally, the MPU-6050 also features a digital low pass filter (DLPF) with configurable bandwidth frequency for both accelerometer and gyroscope. The DLPF is enabled with a bandwidth frequency of approximately 43 Hz.
There is no feedback needed for measuring x and $\dot{x}$ since the motors are open loop controlled.

E. Hardware Architecture

The hardware architecture of the robot electronics is presented in Fig. 3.

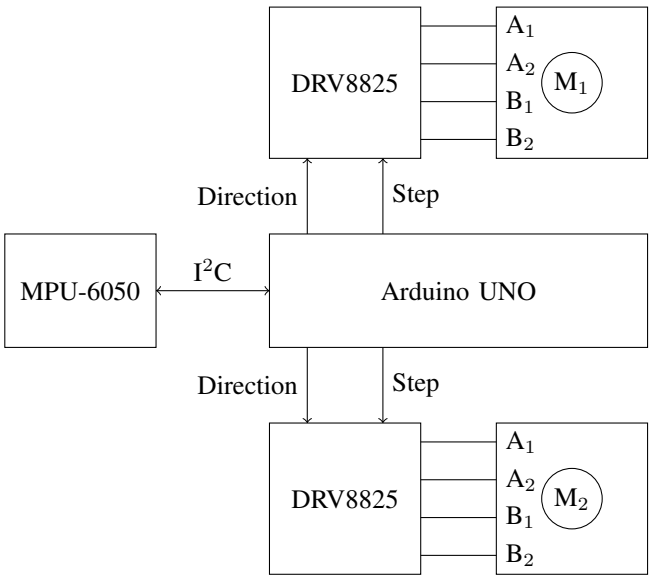


Fig. 3: SBR hardware architecture

The microcontroller is powered using the USB port and the motors are supplied from a 12 V wall adapter. The sensor and the drivers are powered from the microcontroller.

IV. Controller

A. LQR tuning

Using the parameters presented in TABLE I into (II-A.13) the following numerical state space representation of the SBR is obtained

$$\begin{bmatrix}\dot{x}_1\\ \dot{x}_2\\ \dot{x}_3\\ \dot{x}_4\end{bmatrix} = \begin{bmatrix}0 & 1 & 0 & 0\\ 172.0678 & 0 & 0 & 0\\ 0 & 0 & 0 & 1\\ -7.3968 & 0 & 0 & 0\end{bmatrix}\begin{bmatrix}x_1\\ x_2\\ x_3\\ x_4\end{bmatrix} + \begin{bmatrix}0\\ -12.8123\\ 0\\ 1.2812\end{bmatrix} u$$

Using these numerical values an LQR controller is tuned. The selection of the $\mathbf{Q}$ and $\mathbf{R}$ matrices depends on the desired performances. The goal for the SBR presented in this paper is to obtain accurate control in both linear and angular position. An aggressive control for θ and x is then imposed by selecting the following values for $\mathbf{Q}$ and $\mathbf{R}$

$$\mathbf{Q} = \begin{bmatrix}200 & 0 & 0 & 0\\ 0 & 1 & 0 & 0\\ 0 & 0 & 150 & 0\\ 0 & 0 & 0 & 1\end{bmatrix} \qquad \mathbf{R} = 1$$

which yields the following gain vector

$$\mathbf{K} = \begin{bmatrix}-43.6126 & -3.6355 & -12.2474 & -8.7387\end{bmatrix}$$

The tuned controller ensures a small settling time for any non-zero θ and x (considering that for both outputs the set point is 0). This means that only small deviations from the vertical line of the robot's angular position are allowed.

B. Discrete Time Approximation

To implement the control algorithm on a microcontroller a discrete time approximation of the gain vector is computed

$$\mathbf{K}_d = \begin{bmatrix}-39.9624 & -3.3195 & -10.6402 & -7.6361\end{bmatrix}$$

The discrete time values are sampled at a sampling period $T_s = 8ms$, value determined based on the computational speed of the microcontroller.

C. Control Value

The mathematical model presented in subsection II-A and adapted to describe the SBR considers the control effort to be a force measured in Newtons which pulls the robot base forward or backward in order to balance the robot. The closest translation that can be made is to transform linear force into torque. The most common way to control the output torque of a stepper motor is by controlling the stator winding currents by applying the necessary amount of voltage using pulse width modulation. However, the drivers used to control the stepper motors mounted on the real life platform can only be used for speed control. The algorithm for obtaining a speed reference (i.e. obtaining a pulse frequency) from linear force will be described next.
The control effort is split between the two motors driving the robot

$$F_m = \frac{u}{2} \tag{IV-C.1}$$

where F_m is the force generated by one motor. Both of these forces must pull the same mass. The linear acceleration, denoted by a, is obtained as

$$a = \frac{F_m}{M} \tag{IV-C.2}$$

The integral of the linear acceleration gives the linear velocity of the SBR's base.

$$v = \int_0^t a(\tau)d\tau \tag{IV-C.3}$$

A discrete time approximation of the integral used in the implementation of the control algorithm on the microcontroller is given by the following equation

$$v[kT_s] = v[(k-1)T_s] + a[kT_s]T_s \tag{IV-C.4}$$

where k is the sample number. This approximation can lead to an error accumulation but it can be neglected for short running times [14], [15].
Knowing the linear velocity of the robot base, the impulse frequency which is equal to the motor speed measured in steps per second is computed using the following formula

$$f_{step} = \frac{vn}{2\pi r} \tag{IV-C.5}$$

where n is the total number of steps that the motor needs to perform a revolution. A negative value of v denotes a backward rotation of the wheels, while a positive value of v denotes a forward rotation of the wheels.

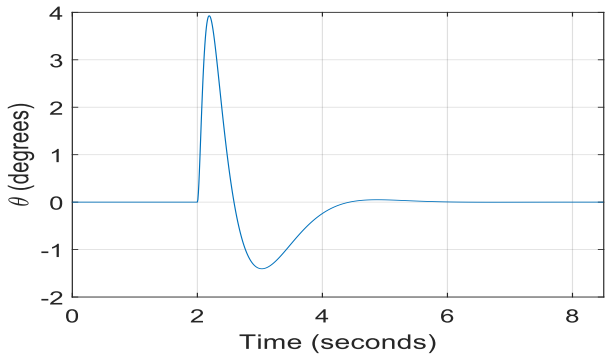
V. Results

A. Numerical and Experimental Data

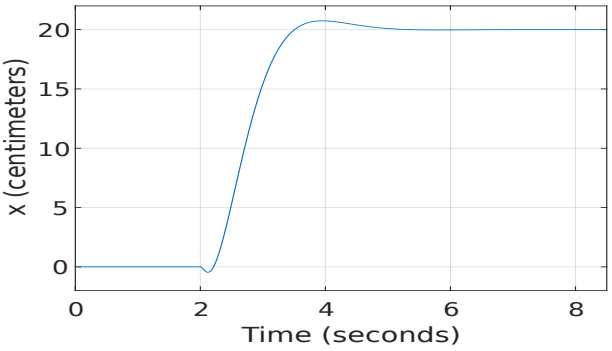
To validate the LQR controller an experimental scenario was first simulated and then performed on the real life platform described in section III.
The robot is positioned vertically at $\theta = 0$ and $x = 0$ and left untouched in this position for two seconds. Then a step reference change of 0.2 equivalent with a displacement of 20 centimeters is applied for x. For a better visualization the angular position is converted to degrees and the linear position is converted to centimeters. The numerical results of the simulated experiment can be seen in Fig. 4, while the experimental results measured on the real life platform in the same scenario are presented in Fig. 5.

B. Discussion

According to the numerical simulation data, from the time moment $t = 0$ until $t = 2$, the robot should not make any movements, staying perfectly balanced in upright position. The same behavior can be observed on the experimental data.
The self-balancing robot has non-minimum phase dynamics which are visible when the step change in the linear position reference is applied. The robot needs to move backward in order to tilt itself in the positive direction which allows it to move forward. Both the simulation and experimental data illustrate the same dynamics.
The transient responses of x in the simulated and experimental

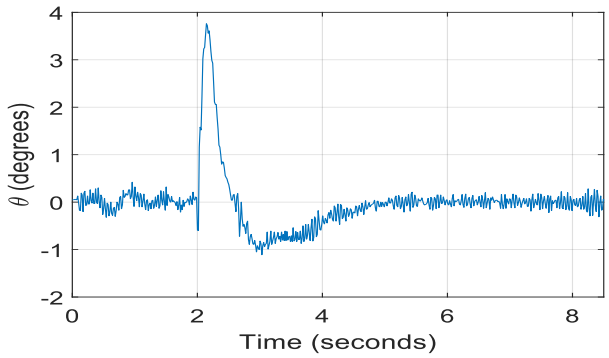


(a) Angular position of the robot

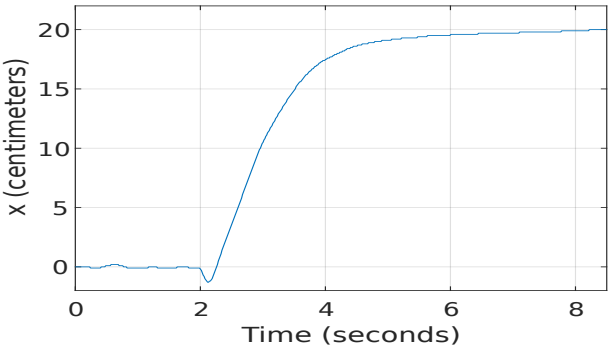


(b) Linear position of the robot

Fig. 4: Simulation Results



(a) Angular position of the robot



(b) Linear position of the robot

Fig. 5: Experimental Results

case differ. The simulation shows a settling time of 2.5 seconds for x with a small overshoot of approximately 3%. On the other hand the response of x on the real life platform has a settling time of 4 seconds with no overshoot.
θ behaves similarly in both the simulated and experimental case. Both plots show underdamped dynamics.The only difference lies in a smaller undershoot of the experimentally measured θ.
The presence of slight differences between the simulated and experimental responses is normal. In the model development phase, several assumptions have been made in order to simplify the modeling task. The real life experimental validation of the LQR control strategy takes into account all the omitted physical variables from the identification step. However, analyzing the results obtained with both the experimental and simulated closed loop systems shows that the obtained self-balancing robot model accurately describes the dynamics of the physical process. The control strategy is validated for both cases. The step change in reference for the linear position acts as a disturbance for the angular position which is successfully rejected. Moreover, the robot changes its linear position to the new set point with no steady state errors.
The experimental results also prove that the approach on the computation of the control value presented in subsection IV-C is correct.

VI. Conclusions

The paper aims to present the complete construction, system modeling and controller development for the case study represented by the self-balancing robot. It includes all the steps from theoretical backgrounds of mathematical modeling and linear quadratic regulator controller to hardware implementation. The results are successfully validated through simulations, as well as real life experiments targeting disturbance rejection and reference tracking performance. The robot features stepper motors as a source of locomotion, providing a different construction option than the already available robots. Another focus of the paper is the tuning of a linear quadratic regulator. The controller is successfully validated on real life data, proving that LQR controllers can be effectively tuned for SBR setups with stepper motors.

References

[1] F. Grasser, A. D'arrigo, S. Colombi, and A. C. Rufer, "Joe: a mobile, inverted pendulum," *IEEE Transactions on industrial electronics*, vol. 49, no. 1, pp. 107–114, 2002.

[2] M. R. Bageant, "Balancing a two-wheeled segway robot," Ph.D. dissertation, Massachusetts Institute of Technology, 2011.

[3] A. A. S. Shaon, S. Bhowmik, B. K. Bhawmick, P. Das, and N. K. Das, "Design and implementation of a self-balancing robot," in *Proc. Int. Conf. on Mechanical Eng. And Renewable Energy*, 2017.

[4] F. Kung, "A tutorial on modelling and control of two-wheeled self-balancing robot with stepper motor," *Applications of Modelling and Simulation*, vol. 3, no. 2, pp. 64–73, 2019.

[5] C. Iwendi, M. A. Alqarni, J. H. Anajemba, A. S. Alfakeeh, Z. Zhang, and A. K. Bashir, "Robust navigational control of a two-wheeled self-balancing robot in a sensed environment," *IEEE Access*, vol. 7, pp. 82 337–82 348, 2019.

[6] M. Okulski and M. Ławryńczuk, "A cascade pd controller for heavy self-balancing robot," in *Conference on Automation*. Springer, 2018, pp. 183–192.

[7] J.-H. Park and B.-K. Cho, "Development of a self-balancing robot with a control moment gyroscope," *International Journal of Advanced Robotic Systems*, vol. 15, no. 2, p. 1729881418770865, 2018.

[8] H. Hellman and H. Sunnerman, "Two-wheeled self-balancing robot: Design and control based on the concept of an inverted pendulum," 2015.

[9] I. Gandarilla, V. Santibáñez, and J. Sandoval, "Stabilization of a self-balancing robot by energy shaping," in *2018 XX Congreso Mexicano de Robótica (COMRob)*. IEEE, 2018, pp. 1–6.

[10] J. Fang, "The lqr controller design of two-wheeled self-balancing robot based on the particle swarm optimization algorithm," *Mathematical Problems in Engineering*, vol. 2014, 2014.

[11] L. B. Prasad, B. Tyagi, and H. O. Gupta, "Optimal control of nonlinear inverted pendulum system using pid controller and lqr: performance analysis without and with disturbance input," *International Journal of Automation and Computing*, vol. 11, no. 6, pp. 661–670, 2014.

[12] M. O. Asali, F. Hadary, and B. W. Sanjaya, "Modeling, simulation, and optimal control for two-wheeled self-balancing robot." *International Journal of Electrical & Computer Engineering (2088-8708)*, vol. 7, no. 4, 2017.

[13] E. Mosca, *Optimal, predictive, and adaptive control*. Prentice Hall Englewood Cliffs, NJ, 1995, vol. 151.

[14] I. R. Birs, C. I. Muresan, S. Folea, and O. Prodan, "An experimental nanomedical platform for controller validation on targeted drug delivery," in *2017 Australian and New Zealand Control Conference (ANZCC)*. IEEE, 2017, pp. 161–165.

[15] I. R. Birs, C. Muresan, D. Copot, I. Nascu, and C.-M. Ionescu, "Identification for control of suspended objects in non-newtonian fluids," *Fractional Calculus and Applied Analysis*, vol. 22, no. 5, pp. 1378–1394, 2019.