Journal of Advanced Engineering Trends
ISSN : 2682 - 2091

Vol.43, No.1. January 2024

http://jaet.journals.ekb.eg

Modeling and Parameters Estimation of a Self-Balancing Two-Wheeled Vehicle

Muhammad Hassan(1) Ahmed Mahmoud Moustafa(2) Mohammed Moness(3)
Instructor Computers and Systems Engineering Dept., Faculty of Engineering, Minia University, Egypt (1)
Assistant Professor Computers and Systems Engineering Dept., Faculty of Engineering, Minia University, Egypt(2)
Professor Computers and Systems Engineering Department, Faculty of Engineering, Minia University, Egypt(3)

A R T I C L E I N F O

Article history:

Received:

Accepted:

Online:

Keywords:

Parameters estimation

Self-Balancing robot

Inverted Pendulum

Two-Wheeled vehicle

A B S T R A C T

The self-balancing two-wheeled vehicle is a practical realization of the well-established control benchmark of the inverted pendulum system. Despite the sophisticated dynamics and kinematics of the inverted pendulum system, it has acquired great interest in real-life applications starting from Segway and hoverboards to the self-balancing wheelchair. These applications benefit from the dynamical structure that provides high maneuverability within narrow spaces. However, the system complexity regarding the high nonlinearity and instability requires accurate models for model-based control of the balancing and motion planning control objectives. This work addresses the modeling and the parameters estimation of a lab-scale version of the self-balancing two-wheeled vehicle. First, a nonlinear dynamical model based on LaGrange kinematics is developed. Then, real-time datasets for closed-loop operations are acquired for offline optimization. Finally, an optimization problem is formulated and solved for the parameter estimation through a decoupled block-by-block approach. The obtained grey-box model is validated for reliable and accurate fitting of the real system. The obtained model was able to simulate the real-time collected data with reasonable meaningful estimated parameters.

1. Introduction

The two-wheeled self-balancing robot (TWSBR) is a naturally unstable, nonlinear system with two independently driven wheels mounted on each side of the robot. The robot's center of mass is located above the wheel axis. Despite its sophisticated dynamics, instability, and nonlinearity inherited from the inverted pendulum system, this robot finds its application in the design and development of control systems for transportation in short distances inside cities such as Segway and hoverboard. It can be considered an environment-friendly mobility solution since it does not produce pollution while having high flexibility and maneuverability in narrow spaces.

Although the TWSBR by nature is an unstable nonlinear system, several investigations and studies have been conducted to model and control this system. An accurate dynamical model is required for a lot of model-based control algorithms such as fuzzy logic control [1], adaptive control [2, 3], and sliding mode control [4]. The TWSBR mathematical model can be obtained using three possible different approaches. The first one is the Newtonian approach [5, 6] where equations of motion derivation is a complicated process, but it gives a natural understanding of the robot movement and relations between forces. The second approach is the LaGrange approach [7, 8] which is the most popular method with multi-body systems. In the Lagrange model, equations of motion can be derived as a function of some generalized coordinates considering the potential and kinetic energy of the system [9, 10]. The last one is Kane's method [11, 12]. Unlike the Newtonian method, the interactive forces do not have to be found. Instead, a generalized force can be determined by the method of partial velocities along generalized coordinates.

Revised:14 June, 2022, Accepted:14 August , 2022

For the mathematical model to accurately simulate the real-time implementation of the TWSBR. Three approaches can be found to obtain the model [13]:

White-box modeling: Assuming a complete knowledge of the systems equations and parameters.

Black-box modeling: Assuming no previous knowledge of systems dynamics. Experimental approaches are carried out to extract the input-output transfer function.

Gray-box modeling: Assuming some knowledge of the system dynamics while the model parameters need to be estimated.

For the white-box modeling method, several studies were carried out with complete knowledge of both the system equations and parameters. In [7], the LaGrange method for the system model is applied by separating it into two parts: The mechanical part representing the robot body, and the actuator part representing the DC motor and the wheels. The model also takes into consideration the energy dissipation of both wheels as they roll along the way and the wheel-surface friction. With some linear approximations, it was managed to use the extracted model for a linear–quadratic regulator (LQR) and full state feedback controller. In [14], a black-box model was obtained by using some closed loop techniques with artificial neural networks. The feedforward neural network was able to predict the tilt angle with a very low mean square error (MSE). Ref [15] considers two phases for system identification. First, the system is considered a multi-input multi-output (MIMO) system where the currents for both motors are the inputs, and the velocities of both wheels are the output. Then, the system is seen as a multi-input single-output (MISO) where the velocities of both wheels (the output of the first phase) are inputs and the tilt angle of the robot is the output. Different model structures were compared such as Autoregressive (AR), Autoregressive Moving Average with Exogenous Inputs (ARMAX), Box-Jenkins (BJ), and Output-Error (OE) as a linear model, and two nonlinear model structures Wiener and Hammerstein. It was concluded that Wiener nonlinear model has the best performance for the first phase with saturation. A wavelet network was suggested to obtain the second phase model. For the gray-box modeling method, ref [16] applied a parameter-dependent time-invariant system with different operating points and scheduled parameters based on the corresponding operating point. It was conducted for a rotary inverted pendulum with considering the numerical condition into account to determine the model structure.

The motivation of this paper is to develop a model for a lab-scale version of the TWSBR since it is a challenging platform to be controlled with a wealth of industrial and practical applications. This paper is structured as follows. The hardware setup is described in Section 2. Then, a mathematical model is developed using LaGrange kinematics in Section 3. In Section 4, an empirically tuned stabilizing PID controller is designed to acquire experiment data. Real-time optimal parameters estimation is carried out and evaluated in Section 5 while Section 5 summarizes conclusions.

2. Hardware construction and configuration

2.1. Mechanical design and construction

The mechanical construction of the robot is separated into two main subsystems. Firstly, the rotational subsystem consists of two actuated wheels driven by two geared DC motors. Secondly, the upper body subsystem consists of three acrylic 3D printed plates, the lower 5mm plate is used to connect the DC motor with the upper body, and the second and the third 3mm plates are used to mount the embedded electronics. **Figure 1** shows the assembled

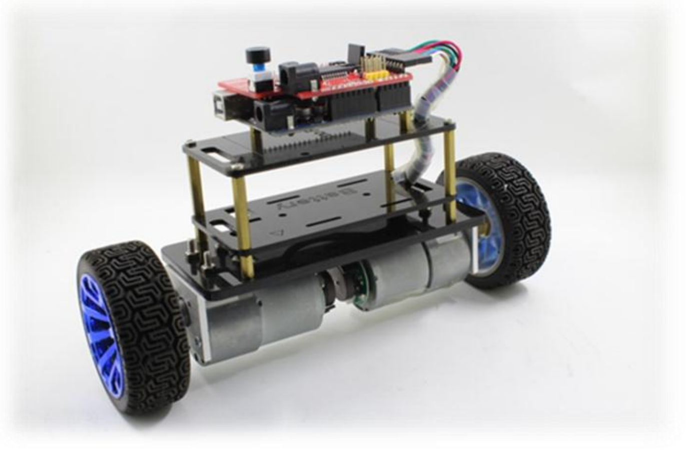

Figure 1: The TWSBR assembled

version of the robot. The dismissions of the robot chassis (length, width, height) are 21.6 x 6.8 x 13.2 cm with a 6.2 cm. wheel diameter. The center of mass of the upper body is located above the wheel axis and the robot has only two contact points with the surface. Because of this mechanical structure, the upper body acts as a pendulum during the translation motion of the robot.

2.2. The embedded system

The TWSBR is built around a small, low-cost, and user-friendly Arduino Uno Rev 3 board which has an ATmega328P microcontroller. A balance shield mounted over the Uno board contains the following:

- *MPU-6050 sensor:* which is, a six degrees of freedom (DoF) inertia measurement unit (IMU) that has an accelerometer and gyroscope. The MPU-6050 is interfaced with the Uno board by an inter-integrated circuit (I2C) bus to provide the measurements of the acceleration and the angular velocity. Furthermore, a Kalman filter is applied to the raw data collected from the MPU-6050 to estimate the robot tilt angle.
- *L298P:* dual H-bridge that drives the two Dc motors.
- *LED indicator:* shows which motor pins are on or off.
- *Motor interface:* that connects the two DC motors to the L298P H-bridge.

The actuators are two high-torque, high-speed DC motors with a gear ratio of 43.8 and two-channel incremental encoders with 16 counts per revolution (CPR) for the motor shaft and 700 CPR for the gear shaft. The motors are driven by a pulse width modulation (PWM) signal supplied to the H-bridge. The embedded electronic system is powered by a 12V – 1A) charger connected through the power input pins. A complete Hardware configuration is depicted in

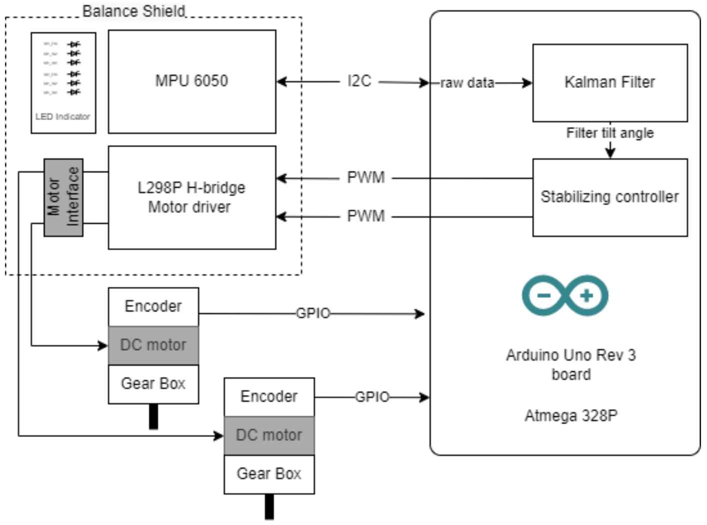

Figure *2*.

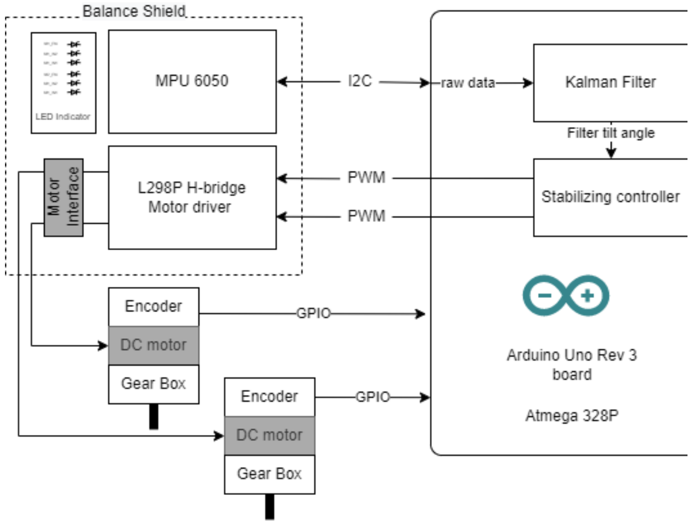

Figure 2: The TWSBR hardware architecture

3. Mathematical model

Accurate modeling with reasonable complexity is essential for model-based control. Some of the model mechanical and electrical parameters are obtained from the datasheets while unknown parameters are experimentally estimated using real time dataset collected from the robot [8].

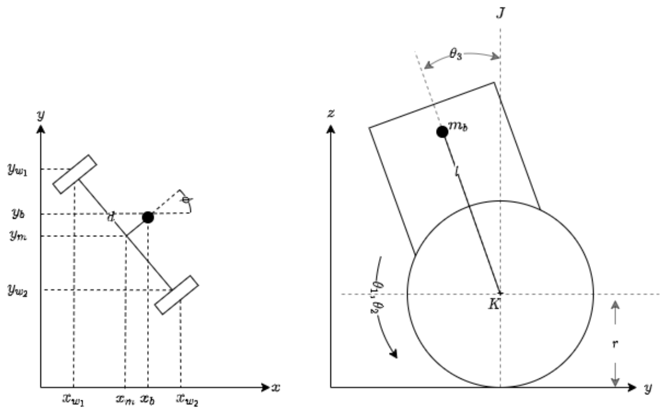

Figure 3: Side and plane view of the TWSBR

Figure 3 shows the kinematic variable for the robot motion $\dot{\theta}_1, \dot{\theta}_2$ indicates the indicates the robot angular displacement for the wheels while θ_3 indicates the robot tilt angle of the upper body.$\dot{\theta}$ indicates the mean value for $\dot{\theta}_1, \dot{\theta}_2$ so that

$$\dot{\theta} = \frac{\dot{\theta}_1 + \dot{\theta}_2}{2} \# (1)$$

While $\dot{\phi}$ is the change in the yaw angle of the robot to indicate the robot direction as:

$$\dot{\phi} = \frac{r(\dot{\theta}_2 - \dot{\theta}_1)}{d} \# (2)$$

$\dot{s} = r\dot{\theta}$ is the linear speed of the robot. The robot parameters are stated in **Table 1**. As indicated in **Figure 3**, the coordinates of the intersection of the two axes J and K (points out of the page) namely x_m, y_m can be obtained as:

$$x_m = \int \dot{s} \cos\phi \, dt \# (3)$$

$$y_m = \int \dot{s} \sin\phi \, dt \# (4)$$

Likewise, the coordinates of both wheels and the upper body can be obtained as follows:

$$x_1 = x_m - \frac{1}{2} d \sin\phi, \qquad y_1 = y_m + \frac{1}{2} d \cos\phi \;\; \# (5)$$

$$x_2 = x_m + \frac{1}{2} d \sin\phi, \qquad y_2 = y_m - \frac{1}{2} d \cos\phi \;\; \# (6)$$

$$z_1 = z_m, \qquad z_2 = z_m \# (7)$$

$$x_b = x_m + l \sin\theta_3 \cos\phi, \qquad y_b = y_m + l \sin\theta_3 \sin\phi \;\; \# (8)$$

$$z_b = z_m - l \cos\theta_3 \# (9)$$

The equations of motion of the system can be derived using the LaGrange approach [17]:

$$\frac{d}{dt} \frac{\partial \mathcal{L}}{\partial \dot{q}_i} - \frac{\partial \mathcal{L}}{\partial q_i} = \tau_i, \qquad i = 1,2, \ldots, N \# (10)$$

where $q = (\theta_1, \theta_2, \theta_3)^T$ is selected to be the generalized coordinates. $\mathcal{L}$ is known as the Lagrange function and is defined as the difference between the kinetic and potential energy, $\mathcal{L} = k - p$ where k is the robot's total kinetic energy which is the sum of both wheel and upper body kinetic energies $k = k_w + k_b$.

p is the robot's total potential energy which is the sum of both wheel and upper body potential energies $p = p_w + p_b$. The kinetic energy of both wheels can be written as:

$$k_w = \sum_{i=1}^{2} \frac{1}{2} m_w (x_i^2 + y_i^2 + z_i^2) + \sum_{i=1}^{2} \frac{1}{2} J_w \theta_i^2 + \sum_{i=1}^{2} \frac{1}{2} n^2 J_r \left(\dot{\theta}_i - \theta_3\right)^2 \ \#(11)$$

where J_w is the wheel's moment of inertia. The upper body kinetic energy is given by:

$$k_b = \frac{1}{2}\left(m_b(\dot{x}_b^2 + \dot{y}_b^2 + \dot{z}_b^2) + J_K \dot{\theta}_3^2 + J_J \dot{\phi}^2\right)\#(12)$$

The robot's potential energy is given by:

$$P = 2m_w g r + m_b(r - l\cos\theta_3)\ \#(13)$$

where g is the gravitational acceleration.

$\tau = [\tau_1, \tau_2, \tau_3]^T$ is the generalized external forces vector in equation (10) which is the difference between the torques generated by the motors and the modeled dissipation due to friction. The motor armature current can be described by the following equations [18]:

$$\dot{I} = \frac{1}{L}\left(u - K_E n \begin{bmatrix} 1 & 0 & -1 \\ 0 & 1 & -1 \end{bmatrix} \dot{q} - RI\right)\#(12)$$

where u is the motor input voltage $u = [u_1, u_2]^T$ and I is the motor armature currents $I = [I_1, I_2]^T$. Equation (12) describes the relationship between the input voltages and the armature current. Since the external torques are proportional to the current equation (13) describes the relationship between the armature current and the motor external torque.

$$\tau_a = K_T n \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ -1 & -1 \end{bmatrix} I\ \#(13)$$

where τ_a is the motor external torques. Furthermore, the modeled friction is composed only of the viscosity friction between the wheel and the surface equation (14) gives the modeled friction.

$$\tau_f = \begin{bmatrix} b_1 + f_{v1} & 0 & -b_1 \\ 0 & b_2 + f_{v2} & -b_2 \\ -b_1 & -b_2 & b_1 + b_2 \end{bmatrix} \dot{q}\ \#(14)$$

The overall generalized external forces can be described as $\tau = \tau_a - \tau_f$.

By evaluating (10) the overall system model can be written as:

$$M(q)\ddot{q} + V(q,\dot{q}) = \tau_a - \tau_f\ \#(15)$$

where $M(q)$ is a 3-by-3 symmetric and positive definite inertia matrix and $V(q,\dot{q})$ is 3-by-1 column vector that describes the potential energy terms. A detailed description of both $M(q)$, and $V(q,\dot{q})$ is given as follows [8]:

$$M(q) = (m_{ij})$$

$$m_{11} = m_{22} = \frac{3}{2} m_w r^2 + \frac{1}{4} m_b r^2 + n^2 J_r + \frac{l^2 r^2}{d^2} m_b \sin^2\theta_3 + J_J \frac{r^2}{d^2}\ \#(16)$$

$$m_{33} = m_b l^2 + J_K + 2n^2 J_r\ \#(17)$$

$$m_{12} = m_{21} = \frac{1}{4} m_b r^2 - \frac{l^2 r^2}{d^2} m_b \sin^2\theta_3 - J_J \frac{r^2}{d^2}\#(18)$$

$$m_{13} = m_{23} = m_{31} = m_{32} = \frac{1}{2} m_b l r \cos\theta_3 - n^2 J_r\ \#(19)$$

$$V(q,\dot{q}) = \begin{bmatrix} \frac{2l^2r^2}{d^2} m_b \sin\theta_3 \cos\theta_3 \dot{\theta}_3 \left(\dot{\theta}_1 - \dot{\theta}_2\right) - \frac{1}{2} m_b l r \sin\theta_3 \dot{\theta}_3 \\ \frac{2l^2r^2}{d^2} m_b \sin\theta_3 \cos\theta_3 \dot{\theta}_3 \left(\dot{\theta}_2 - \dot{\theta}_1\right) - \frac{1}{2} m_b l r \sin\theta_3 \dot{\theta}_3 \\ -\frac{l^2r^2}{d^2} m_b \sin\theta_3 \cos\theta_3 \left(\dot{\theta}_1 - \dot{\theta}_2\right)^2 + m_b g l \sin\theta_3 \end{bmatrix}\#(20)$$

The state-space representation of the model of the overall system can be stated as follows:

$$\dot{x}(t) = \begin{bmatrix} \dot{q} \\ M(q)^{-1}\left(\tau_a - \tau_f - V(q,\dot{q})\right) \\ \frac{1}{L}\left(u - K_E n \begin{bmatrix} 1 & 0 & -1 \\ 0 & 1 & -1 \end{bmatrix} \dot{q} - RI\right) \end{bmatrix}\#(21)$$

$$y(t) = x(t)\ \#(22)$$

Table 1: TWSBR parameter estimation

Symbol	Parameter name	Initial values [SI Unit]
r	Wheel radius	3.25×10^{-2}
m_w	Wheel mass	53×10^{-3}
l	Distance between the center of gravity of the upper body and wheel axle	4.5×10^{-2}
m_b	Mass of the upper body	0.8
d	The between the two wheels	16.5×10^{-2}
J_J	Moment of inertia of the robot about the J axis	55.18×10^{-4}
J_K	Moment of inertia of the robot about the K axis	13.50×10^{-4}
n	Gear ratio	43.8
J_r	The inertia of the rotor	2.74×10^{-6}
R	The rotor resistance	4.019
L	The rotor inductance	2.22×10^{-5}
K_T	Torques constants	54.43×10^{-4}
K_E	Back-EMF constants	54.43×10^{-4}
b	Viscous friction of the motor	7.27×10^{-6}
f_v	Viscous friction of the wheel	0.54×10^{-3}

4. Optimal parameter estimation

4.1. Dataset acquisition

Since the TWSBR is unstable, a stabilizing controller is required for the offline dataset acquisition to estimate the model parameters. An empirically tuned PID is implemented to stabilize the robot. With the help of the Arduino support package, and the Embedded Hardware coder for Simulink the controller is realized using Simulink blocks. The control algorithm and the real-time model run with a sample time of $0.005\ s$. The tunned controller is mainly used for stabilizing the robot for the offline data set collection not to fully control the robot. The only design requirement for the controller is to stabilize the robot around the vertical equilibrium point. A stabilizing controller can be obtained through the following algorithm.

1. Start by setting the proportional gain $K_P = 10$, setting both $K_I, K_D = 0$. The robot is unable to recover with very low K_P.
2. Increase K_P by a factor of 10 until $K_P = 60$. The robot begins to jitter too much. An acceptable compromise between the two values is $K_P = 25$ where the robot can balance with a little external help.
3. Begin to tune K_D value by setting the $K_D = 1, K_P = 25, K_I = 0$. The robot now can balance with less aggressive recovery behavior.
4. Increasing the K_D will make the robot jitter again. A suitable value is $K_D = 3.5$ can make the robot recover with less aggressive behavior.
5. Finally, tune the K_I parameter to remove jittering and help the robot to balance with a smoother behavior. The tuned value is $K_I = 3$.

The real-time model was compiled and deployed using the Simulink embedded coder and hardware support packages.

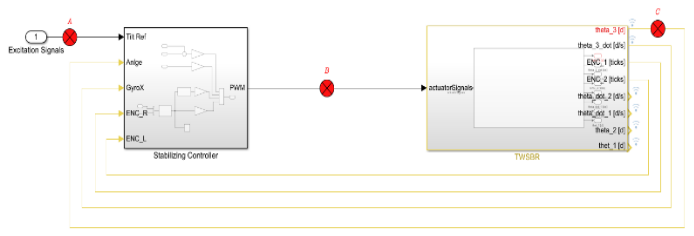

Figure 4: TWSBR closed-loop Simulink model for real-time deployment

Figure 4 indicates the closed-loop model for the TWSBR (overall model). While **Figure 5** indicates a detailed implementation of the stabilizing controller.

There exist three points marked with red in **Figure 4**. The data set is collected between the input of the TWSBR (input voltage- point B), and the robot states vector that includes tilt angle, angular velocity, wheel speed, and wheel displacement (point C). While point A indicates the input of the excitation signals. Several well-known signals are used to perturb the robot and collect the robot states. The control signal generated by the stabilization controller indicated by the output port (1) named PWM in **Figure 5** is the pulse width modulation signal supplied to the H-bridge through the Arduino analog pins.

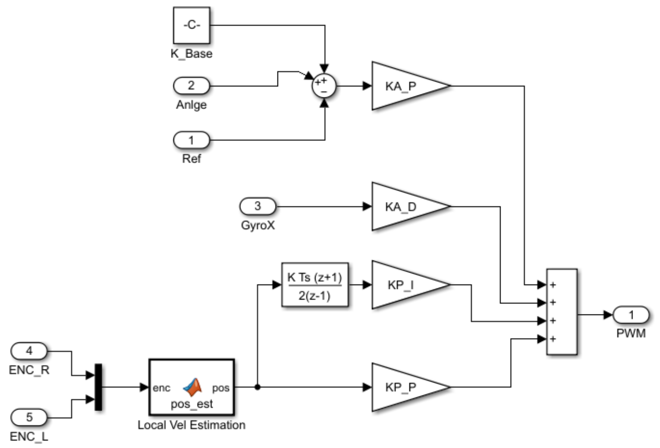

Figure 5: Detailed implementation of the stabilizing controller

4.2. Parameter estimation

The numerical simulation of the mathematical model was conducted using the MATLAB Simulink model. Where the state-space model in (21) is developed with the help of Simulink S-function. **Figure 6** shows the developed Simulink model.

Four experiments were carried out to estimate the model parameters. One experiment is used for model parameter estimation, while the other three were used for model validation.

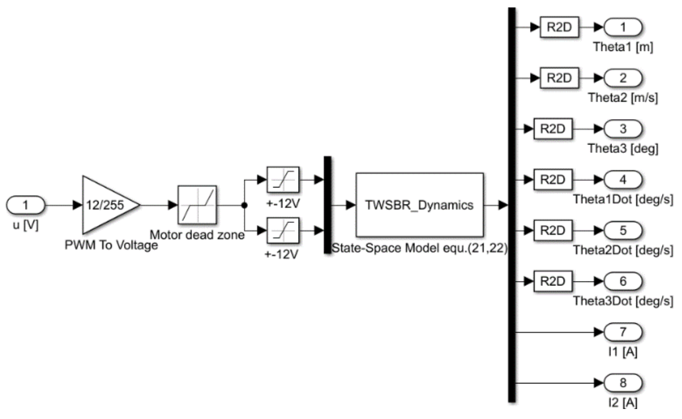

Figure 6:Numerical simulation of state-space representation using Simulink S-function for offline estimation.

After dataset acquisition and with the help of the Simulink parameter estimator app, an optimization problem is formulated. The active-set algorithm focuses on solving Karush-Kuhn-Tucker (KKT) equations. The KKT equations are necessary conditions for optimality in a constrained problem. For a general optimization problem (GP) to find a set of design parameters $x = (x_1, x_2, x_3, \dots\dots, x_n)$ is stated as follows:

$$\min f(x) \#(23)$$

Subject to

$$G_i(x) = 0, i = 1, \dots, m_e \#(24)$$

$$G_i(x) \le 0, i = m_e + 1, \dots, m \#(25)$$

$$x_l \le x \le x_u, \#(26)$$

Where x is the vector of length n design parameters, $f(x)$ is the objective function, and $G(x)$ is a vector of length m contains the equality and inequality constraints.

While the KKT equations can be stated as follows:

$$\nabla f(x^*) + \sum_{i=1}^{m} \lambda_i . \nabla G_i(x^*) = 0 \#(27)$$

$$\lambda_i . G_i(x^*) = 0, i = 1, \ldots, m_e \#$$

$$\lambda_i \geq 0, i = m_e + 1, \ldots, m \#28$$

The cost function was chosen to be the linear sum of squared errors (SSE) which can be stated as follows:

$$SSE = \sum_{i=1}^{n} (y_i - \widehat{y_i})^2$$

where y_i is the real-time data points and $\widehat{y_i}$ the model simulation data points. To have physically meaningful optimal estimates, the estimated parameters were constrained to the values in Table 2. While the convergence of the algorithm is in **Figure 8** showing a scaled version of the error function indicating how the cost function decreases as the number of iterations increase the algorithm converged in twelve iterations. All model parameters were chosen to be estimated to give the model the ability to reach a local minimum without overfitting the collected data. In addition, the sensitivity analysis for all parameters is shown in **Figure 7** indicating the correlation of each estimated parameter to affect the simulation model.

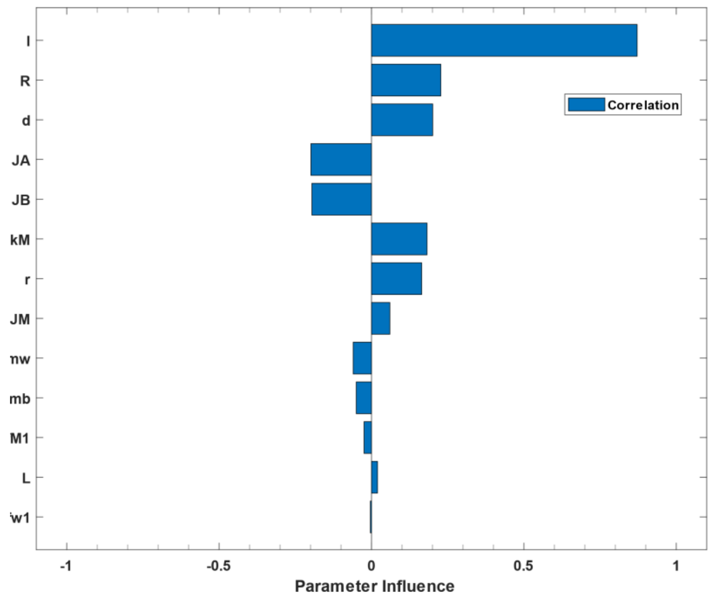

Figure 7 Sensitivity analysis for model parameters

Figure 9 shows the first experiment where the model is excited by a square wave with an amplitude of 8 and frequency of 5 rad/s. **Figure 10** shows the model simulation for the tunned parameters, as it can be seen the model is accurately fitting the tracking the real-time data after the parameters are tuned. **Table 3**shows the tuned parameter values. Consequently, the model is further validated using the other three experiments. **Figure 11** shows the validated model doing reasonably good performance simulating the real-time measurements. **Figure 12** shows the model response to a doublet function the model was able to track the excitation signal. While **Figure 13** shows the model response to a sine stream signal with different frequencies 0.5, 1 Hz and amplitudes 8.5 $degrees$.

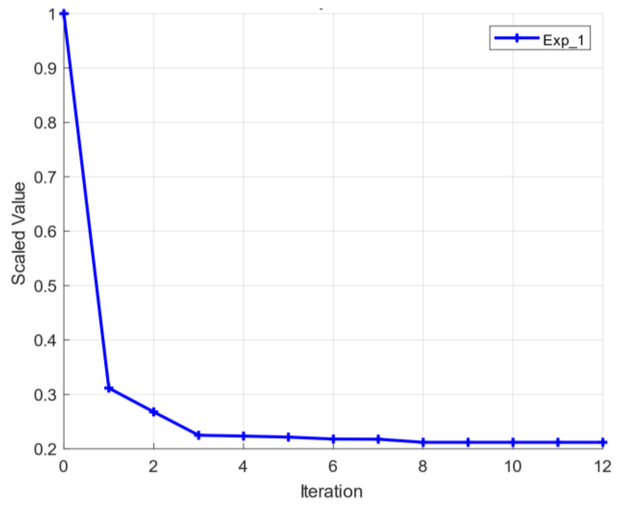

Figure 8 Convergence behavior of the cost function

Table 2 Estimated parameters constrains

Symbol	Minimum values [SI units]	Maximum values [SI units]
r	0.03	0.04
m_w	0.04	0.06
l	0	0.2
m_b	0.3	1.5
d	0.15	0.17
J_J	$5 * 10^{-5}$	0.02
J_K	$1 * 10^{-5}$	0.1
J_r	$2 * 10^{-7}$	$3 * 10^{-3}$
R	1	20
L	$2 * 10^{-7}$	0.002
K_T, K_E	$5 * 10^{-5}$	0.5
b	$7 * 10^{-7}$	$7 * 10^{-3}$
f_v	$5 * 10^{-6}$	0.05

Table 3: TWSBR tuned parameters

Symbol	Parameter name	Values [SI Unit]
r	Wheel radius	0.03015
m_w	Wheel mass	0.0585
l	Distance between the center of gravity of the upper body and wheel axle	0.0489
m_b	Mass of the upper body	0.585
d	The between the two wheels	0.1625
J_J	Moment of inertia of the robot about the J axis	0.0189
J_K	Moment of inertia of the robot about the K axis	0.00750
n	Gear ratio	43.8
J_r	The inertia of the rotor	$1.538 * 10^{-5}$
R	The rotor resistance	5.5609
L	The rotor inductance	$23.68 * 10^{-6}$
K_T	Torques constants	0.00586
K_E	Back-EMF constants	0.00586
b	Viscous friction of the motor	$8.153 * 10^{-6}$
f_v	Viscous friction of the wheel	0.00642

5. Conclusion

The TWSBR is an unstable system that has many academic and industrial applications. Low-cost Lab-Scale TWSBR was constructed using Arduino Uno REV3 embedded board. Then, a mathematical model was developed using the LaGrange kinematics. A dataset is collected from the robot stabilizing in a closed-loop configuration with an empirically tuned PID. An optimization problem is formulated and solved to estimate the model parameters such that the model matches the real-time measurement accurately. The estimated parameters were meaningful for the physical system such that the estimated parameters were close the measured parameters.

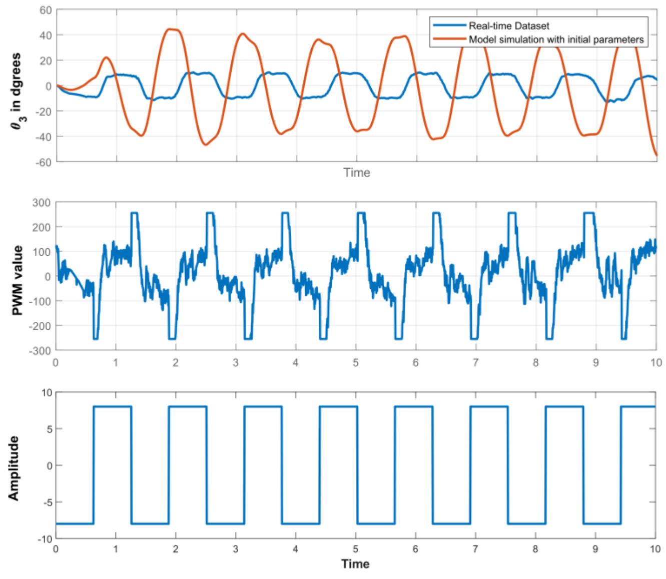

Figure 9: Model simulation with initial set of parameters

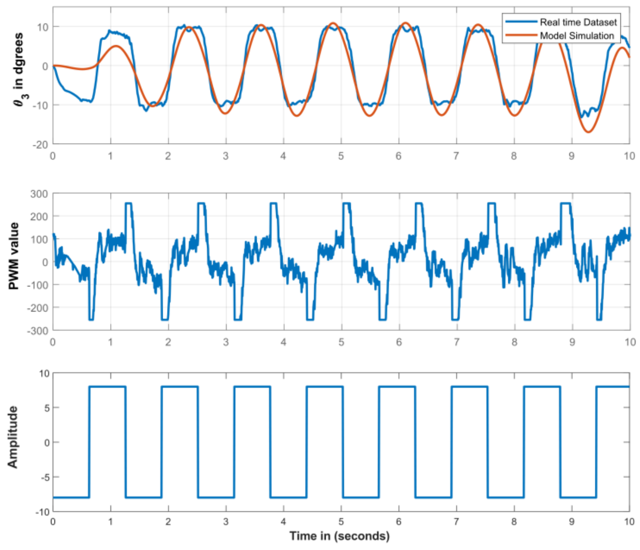

Figure 10: Model simulation with tuned parameters

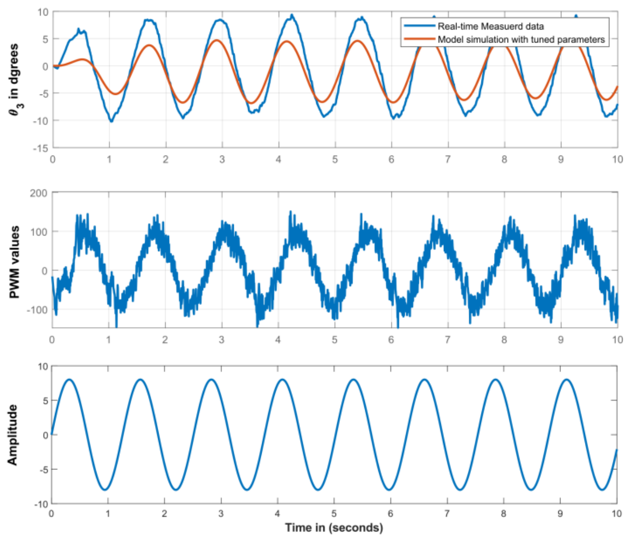

Figure 11: Model Response to a sin wave signal with the tuned parameters

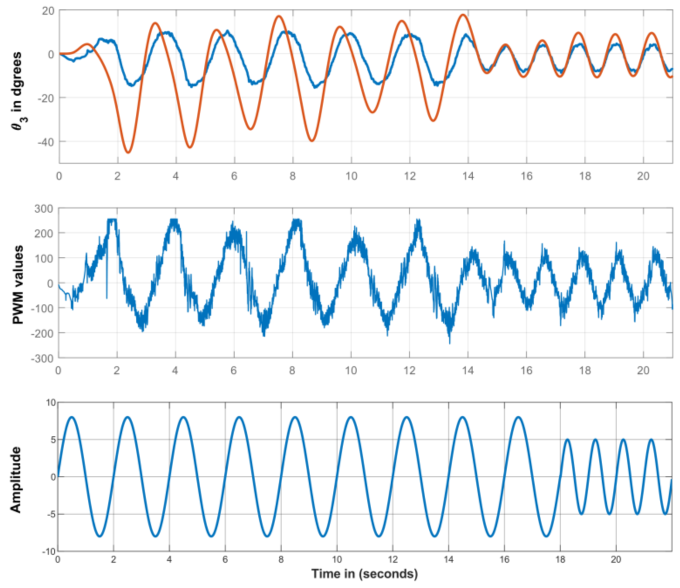

Figure 13 Model response to a Sin stream input with tuned parameters

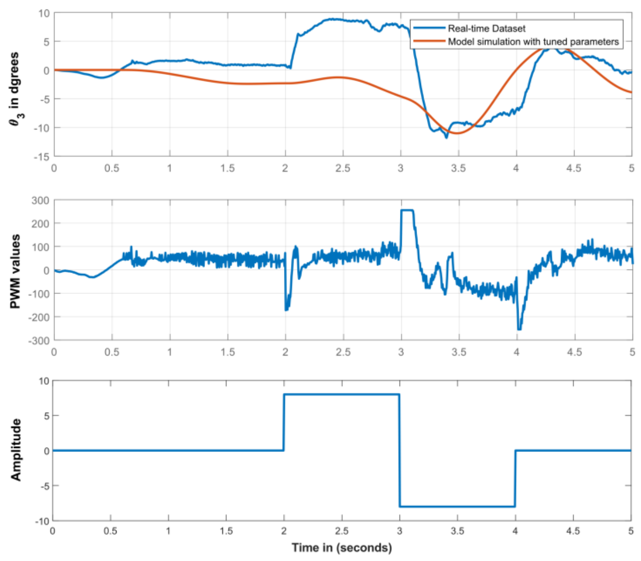

Figure 12: Model response to a doublet input with tuned parameters

References

[1] Á. Odry, E. Burkus, I. Kecskés *et al.*, "Fuzzy control of a two-wheeled mobile pendulum system." pp. 99-104.

[2] Z. Li, and J. Luo, "Adaptive Robust Dynamic Balance and Motion Controls of Mobile Wheeled Inverted Pendulums," *IEEE Transactions on Control Systems Technology*, vol. 17, no. 1, pp. 233-241, 2009.

[3] S.-C. Lin, C.-C. Tsai, and H.-C. Huang, "Adaptive Robust Self-Balancing and Steering of a Two-Wheeled Human Transportation Vehicle," *Journal of Intelligent & Robotic Systems*, vol. 62, no. 1, pp. 103-123, 2011.

[4] J. Huang, Z. H. Guan, T. Matsuno *et al.*, "Sliding-Mode Velocity Control of Mobile-Wheeled Inverted-Pendulum Systems," *IEEE Transactions on Robotics*, vol. 26, no. 4, pp. 750-758, 2010.

[5] J. Li, X. Gao, Q. Huang *et al.*, "Mechanical Design and Dynamic Modeling of a Two-Wheeled Inverted Pendulum Mobile Robot." pp. 1614-1619, 2007.

[6] C. Huang, W. Wang, and C. Chiu, "Design and Implementation of Fuzzy Control on a Two-Wheel Inverted Pendulum," *IEEE Transactions on Industrial Electronics*, vol. 58, no. 7, pp. 2988-3001, 2011.

[7] P. Frankovsky, L. Dominik, A. Gmiterko *et al.*, "Modeling of two-wheeled self-balancing robot driven by DC gearmotors," *International Journal of Applied Mechanics and Engineering*, vol. 22, no. 3, 2017.

[8] Á. Odry, I. Harmati, Z. Király *et al.*, "Design, realization and modeling of a two-wheeled mobile pendulum system." pp. 75-79, 2015.

[9] F. Dai, X. Gao, S. Jiang *et al.*, "A two-wheeled inverted pendulum robot with friction compensation," *Mechatronics*, vol. 30, pp. 116-125, 2015.

[10] Y.-S. Ha, “Trajectory tracking control for navigation of the inverse pendulum type self-contained mobile robot,” *Robotics and autonomous systems,* vol. 17, no. 1-2, pp. 65-80, 1996.

[11] T. Takei, R. Imamura, and S. Yuta, “Baggage Transportation and Navigation by a Wheeled Inverted Pendulum Mobile Robot,” *IEEE Transactions on Industrial Electronics,* vol. 56, no. 10, pp. 3985-3994, 2009.

[12] T. R. Kane, and D. A. Levinson, *Dynamics, theory and applications*: McGraw Hill, 1985.

[13] M. Moness, and A. M. Mostafa, “An algorithm for parameter estimation of twin-rotor multi-input multi-output system using trust region optimization methods,” *Proceedings of the Institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering,* vol. 227, no. 5, pp. 435-450, 2013.

[14] P. Gautam, "System identification of nonlinear Inverted Pendulum using artificial neural network." pp. 1-5.

[15] M. Shahraki, M. A. Sh, and A. Mousavinia, "Two wheel self-balanced mobile robot identification based on experimental data." pp. 883-888.

[16] H. Tanaka, and Y. Ohta, “Grey-box Modeling of an Inverted Pendulum System Based on PD-LTI System,” *Transactions of the Institute of Systems, Control and Information Engineers,* vol. 29, no. 12, pp. 544-551, 2016.

[17] J. Zhang, T. Zhao, B. Guo *et al.*, “Fuzzy fractional-order PID control for two-wheeled self-balancing robots on inclined road surface,” *Systems Science & Control Engineering,* vol. 10, no. 1, pp. 289-299, 2022/12/31, 2022.

[18] A. A. Aldhalemi, A. A. Chlaihawi, and A. Al-Ghanimi, “Design and Implementation of a Remotely Controlled Two-Wheel Self-Balancing Robot,” *IOP Conference Series: Materials Science and Engineering,* vol. 1067, no. 1, pp. 012132, 2021/02/01, 2021.