ORIGINAL SCIENTIFIC PAPER OPEN ACCESS

ADP–based control of a two–wheeled self–balancing mobile robot

Vladimir Stojanović[1*], Vladimir Djordjević[1], Ljubiša Dubonjić[1], Marko Nikolić[1]

[1] University of Kragujevac, Faculty of Mechanical and Civil Engineering, Kraljevo, Serbia

ARTICLE INFO

[*] **Correspondence:** vladostojanovic@mts.rs

DOI: 10.5937/engtoday2400018S

UDC: 621(497.11) **ISSN:** 2812-9474

Article history: Received 5 November 2024; Revised 10 December 2024; Accepted 26 December 2024

ABSTRACT

This study investigates optimal control for a two-wheeled, self-balancing mobile robot whose dynamics is unknown. The objective is to achieve asymptotic control and rejection of disturbances while minimizing a certain index of performances. An approximate optimum controller may be iteratively learnt online utilizing quantifiable input and output data by combining the internal model concept with adaptive dynamic programming (ADP). States that cannot be measured directly are additionally recreated using this information. The ADP method is used to solve the discrete-time algebraic Riccati problem iteratively. The efficacy and viability of the suggested approach are shown by the simulation results.

KEYWORDS

Adaptive dynamic programming, Adaptive optimal control, Two-wheeled self-balancing mobile robot, Unknown dynamics, Algebraic Riccati equation.

1. INTRODUCTION

One well-known example of a robot having potential uses in a variety of fields, including research and transport, is a self-balancing mobile robot. The control of a self-balancing robot is gaining significant interest in both academics and industry. Considering this kind of robot is intrinsically unstable, high-order, multivariable, nonlinear, and highly coupled, it is a mechanical system that is especially underactuated. Dynamic coupling is necessary to indirectly manage the unactuated generalized coordinates since an underactuated system has less control inputs than the generalized coordinates. Underactuation presents difficulties for controller design even if it results in less actuators, which can minimize the production expenses and lower rates of failure. Furthermore, a self-balancing robot moves along its path while keeping the pendulum balanced, in contrast to simpler systems like an inverted pendulum on a cart, which is restricted to a set track. Controlling a self-balancing robot's linear, tilt, and yaw motions all at once is one of the difficulties.

Adaptive dynamic programming (ADP) presents a practical and efficient strategy for employing either traditional or intelligent control methods to get the best possible control performances. It combines the principles of dynamic programming with neural networks, aiming to resolve optimal control challenges in dynamic programming problems by leveraging the approximation capabilities of neural networks [1,2]. ADP has recently been broadened and implemented in various fields, involving robots, spaceships, and more [3,4].

Utilizing output feedback-based ADP is appropriate in situations when the status of a system cannot be directly measured and its matrices are unknown. The study mentioned in [5] introduces an output feedback ADP method for discrete-time linear systems. This method uses measurable input and output data to define the states of the discretized

model, and it follows with policy iteration (PI) and value iteration (VI) to derive the optimal control policy. However, to ensure a unique solution during each iteration, it is necessary to introduce some exploration noise, which may impact the result's precision.

Direct measurement of system states is frequently too costly. Approaches to self-applied state estimation assume that parameters of systems don't change. However, in reality, these criteria cannot be known. The dynamic behavior of complex systems may also be described by linear stochastic state-space models with online approximated dynamics [6]. An exact understanding of system characteristics and states is critical for successfully implementing various control strategies. Many actual engineering applications, such as intelligent vehicles [7], robotic manipulation tasks [8], and 2-degree-of-freedom helicopters [9], need the use of linear models for real-time control.

The optimal control issue is then tackled after the continuous-time linear plant is transformed into a discrete form for simpler implementation in reality. The discrete model is used for controlling a self-balancing mobile robot with uncertain dynamics using an adaptive optimum output feedback approach. Simulation findings show that the suggested control strategy is valid and effective, with exploration noise not affecting the accuracy of the discrete Riccati equation's solution.

2. DESCRIPTION OF THE MOBILE ROBOT

One type of multivariable underactuated mechanical system is a self-balancing robot [10]. The total amount of generalized coordinates in underactuated mechanical systems is more than the total number of actuators. In these systems, the generalized coordinates are controlled indirectly through their interconnection. Also, a self-balancing robot is a high-order, nonlinear and unstable system. The mobile robot consists of a chassis and attached wheels driven by electric motors, see Fig. 1. The robot has three degrees of freedom (generalized coordinates):

1) tilt angle θ (rotation around the Z-axis)
2) linear motion along the X-axis
3) yaw angle δ (rotation around the Y-axis)

These three generalized coordinates and their corresponding velocities fully describe the dynamics of the robot and represent the elements of the state vector.

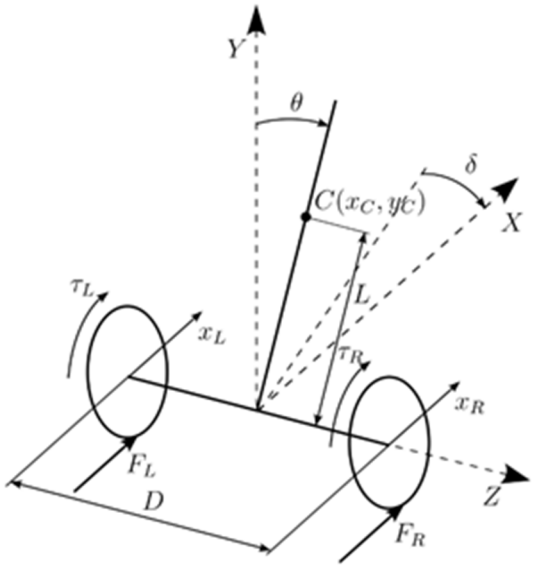

Figure 1: A self-balancing mobile robot

The robot is controlled by two electric motors that drive the wheels with torques τ_L and τ_R. Other parameters of the robot are: $M[\mathrm{kg}]$ - Mass of the chassis; $m[\mathrm{kg}]$ - Each wheel's mass; $R[\mathrm{m}]$ - Wheel radius; $D[\mathrm{m}]$ - The two wheels' distance; $L[\mathrm{m}]$ - Length from the robot's center of gravity to the Z-axis; $J_y\left[\mathrm{kg}\cdot\mathrm{m}^2\right]$ - Moment of inertia of the chassis with respect to the Y-axis; $J_z\left[\mathrm{kg}\cdot\mathrm{m}^2\right]$ - Moment of inertia of the chassis with respect to the Z-axis; $J_\omega\left[\mathrm{kg}\cdot\mathrm{m}^2\right]$ - The wheel's moment of inertia in relation to the Z-axis; x_L, $x_R[\mathrm{m}]$ - Wheel displacements on the left and right; x_C, $y_C[\mathrm{m}]$ - The location of the robot's center of gravity; X_L, $X_R[\mathrm{N}]$ - Forces that interact on the X-axis among the wheels and the frame; Y_L, $Y_R[\mathrm{N}]$ - Interactions among the wheels and the frame on the Y-axis; F_L, $F_R[\mathrm{N}]$ - Friction forces among

the wheels and the hard surfaces; τ_L, $\tau_R[\mathrm{N\cdot m}]$ - Torque produced by the motors on the left and right; θ_L, $\theta_R[\mathrm{rad}]$ - Angles at which the left and right wheels rotate. The dynamics of the wheels is defined by the sum of forces:

$$m\ddot{x}_L = F_L - X_L \tag{1}$$

$$m\ddot{x}_R = F_R - X_R \tag{2}$$

and totals of moments

$$J_\omega \ddot{\theta}_L = \tau_L - F_L R \tag{3}$$

$$J_\omega \ddot{\theta}_R = \tau_R - F_R R \tag{4}$$

The dynamics of the chassis is described by the sums of the forces for the x and y axes, as well as the sums of the moments for the z and y axes as follows:

$$M\ddot{x}_C = X_L + X_R \tag{5}$$

$$M\ddot{y}_C = Y_L + Y_R - Mg \tag{6}$$

$$J_z \ddot{\theta} = (Y_L + Y_R)L\sin(\theta) - (X_L + X_R)L\cos(\theta) - (\tau_L + \tau_R) \tag{7}$$

$$J_y \ddot{\delta} = \frac{D}{2}(X_L - X_R) \tag{8}$$

If we assume no wheel slip, then $x_L = R\theta_L$ and $x_R = R\theta_R$. Based on this, we can solve equations (1)-(4) and eliminate from them the angles of rotation of the wheels θ_L and θ_R, as well as the friction forces on the wheels F_L and F_R. As a result, we get the following equations that describe the dynamics of the left and right wheel:

$$\left(M + \frac{J_\omega}{R^2}\right)\ddot{x}_L = \frac{\tau_L}{R} - X_L \tag{9}$$

$$\left(M + \frac{J_\omega}{R^2}\right)\ddot{x}_R = \frac{\tau_R}{R} - X_R \tag{10}$$

The connection between the robot's yaw angle δ and the wheels' linear motion x_L and x_R is given by the expression:

$$D\delta = x_L - x_R \tag{11}$$

The center of mass of the robot is determined by coordinates

$$x_C = x + L\sin(\theta) \tag{12}$$

$$y_C = L\cos(\theta) \tag{13}$$

where

$$x = \frac{1}{2}(x_L + x_R) \tag{14}$$

Based on the above equations, we can get the system's nonlinear equations

$$\ddot{x}\left(M + 2m + \frac{2J_\omega}{R^2}\right) + ML\left(\ddot{\theta}\cos(\theta) - \dot{\theta}^2\sin(\theta)\right) = \frac{1}{R}(\tau_L + \tau_R) \tag{15}$$

$$\left(\frac{2J_y}{D} + \frac{DJ_\omega}{R^2} + Dm\right)\ddot{\delta} = \frac{1}{R}(\tau_L - \tau_R) \tag{16}$$

$$\begin{aligned} J_z\ddot{\theta} = {} & 2\ddot{x}L\left(m + \frac{J_\omega}{R^2}\right)\cos(\theta) + MgL\sin(\theta) - ML^2\ddot{\theta}\sin^2(\theta) - \\ & - ML^2\dot{\theta}^2\sin(\theta)\cos(\theta) - \left(1 + \frac{L\cos(\theta)}{R}\right)(\tau_L + \tau_R). \end{aligned} \tag{17}$$

Making these nonlinear equations linear around the operating point $\theta = 0$, we derive the subsequent linear state space representation of the self-balancing mobile robot

$$\begin{bmatrix} \dot{x} \\ \ddot{x} \\ \dot{\theta} \\ \ddot{\theta} \\ \dot{\delta} \\ \ddot{\delta} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & a_{23} & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & a_{43} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \\ \theta \\ \dot{\theta} \\ \delta \\ \dot{\delta} \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ b_{21} & b_{22} \\ 0 & 0 \\ b_{41} & b_{42} \\ 0 & 0 \\ b_{61} & b_{62} \end{bmatrix} \tag{18}$$

$$y = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \\ \theta \\ \dot{\theta} \\ \delta \\ \dot{\delta} \end{bmatrix} \tag{19}$$

where $\begin{bmatrix} x & \dot{x} & \theta & \dot{\theta} & \delta & \dot{\delta} \end{bmatrix}^T$ is a state vector, $\begin{bmatrix} \tau_L & \tau_R \end{bmatrix}^T$ represents a vector input, $\begin{bmatrix} x & \theta & \delta \end{bmatrix}^T$ represents an output vector, while the model parameters are defined as

$$a_{23} = \frac{-M^2L^2g}{MJ_z + 2\left(J_z + ML^2\right)\left(m + J_\omega/R^2\right)}$$

$$a_{43} = \frac{M^2gL + 2MgL\left(m + J_\omega/R^2\right)}{MJ_z + 2\left(J_z + ML^2\right)\left(m + J_\omega/R^2\right)}$$

$$b_{21} = b_{22} = \frac{\left(J_z + ML^2\right)/R + ML}{MJ_z + 2\left(J_z + ML^2\right)\left(m + J_\omega/R^2\right)}$$

$$b_{41} = b_{42} = \frac{-(R+L)M/R - 2\left(m + J_\omega/R^2\right)}{MJ_z + 2\left(J_z + ML^2\right)\left(m + J_\omega/R^2\right)}$$

$$b_{61} = -b_{62} = \frac{D/2R}{J_y + D^2/(2R)\left(mR + J_\omega/R\right)}$$

3. OPTIMAL PROBLEM FORMULATION

For practical application in a self-balancing mobile robot's control system, we shall explore the discrete system expressed in

$$x_{k+1} = A_d x_k + B_d u_k \tag{20}$$

$$y_k = C x_k \tag{21}$$

in which $A_d = e^{Ah}$, $B_d = \int_0^h \left(e^{A\tau} d\tau\right) B$ and $h > 0$ is the time frame for sampling, In the event $\omega_h = 2\pi/h$ is non-pathological rate of sampling [11]. That is to say, no two eigenvalues of A in which both imaginary and real elements are identical differ by an integral multiple of ω_h. The input, output, and state vectors at the moment of sampling kh are x_k, u_k, y_k, respectively. After that, both $\left(A_d, C\right)$ and $\left(A_d, Q^{1/2}C\right)$ are observable and $\left(A_d, B_d\right)$ is controllable. The cost for (20)-(21) is:

$$J_d(x_0) = \sum_{j=0}^{\infty} y_j^T Q y_j + u_j^T R u_j \ . \tag{22}$$

The law of optimal control that minimizes (22) is

$$u_k = -K_d^* x_k \, , \tag{23}$$

where discrete optimal feedback gain matrix is $K_d^* = \left(R + B_d^T P_d^* B_d\right)^{-1} B_d^T P_d^* A_d$, and P_d^* is the unique symmetric positive definite solution to

$$A_d^T P_d^* A_d - P_d^* + C^T Q C - A_d^T P_d^* B_d K_d^* = 0\,. \tag{24}$$

So far, only low-order simple linear systems have been able to use this well-known optimum control design technique. In reality, for systems of a high order and size, it is frequently challenging to straight away resolve P_d^* from (24), which is nonlinear in P_d. Nonetheless, numerous effective techniques have been created to numerically estimate the solution of (24). A certain algorithm has been established by Hewer [12]. Through the iterative solution of the Lyapunov equation

$$\left(A_d - B_d K_j\right)^T P_j \left(A_d - B_d K_j\right) + C^T Q C + K_j^T R K_j = 0 \tag{25}$$

which is linear in P_j, and updating K_j through

$$K_j = \left(R + B_d^T P_{j-1} B_d\right)^{-1} B_d^T P_{j-1} A_d \tag{26}$$

A numerical approximation of the nonlinear equation (24) solution is provided. According to the results of this approach, sequences $\left\{P_j\right\}_{j=0}^{\infty}$ and $\left\{K_j\right\}_{j=0}^{\infty}$ converge to P_d^* and K_d^*, respectively. In addition, for $j = 0,1,\ldots$, $A_d - B_d K_j$ is a Schur matrix.

It should be noted that Hewer's approach is a model-based policy iteration (PI) technique, which means it cannot be used in situations when all of the matrices of the system are unknown. This method relies on system parameters and operates as an offline method. We will create an output feedback adaptive optimum control method for the discrete system (20) – (21) that does not need knowledge of the matrices of the system in order to use it online.

4. DESIGN OF OPTIMAL ADAPTIVE CONTROL

Similarly to [5], it is possible to express the expanded state equation using inputs and outputs on a time frame as $[k-N, k-1]$:

$$\begin{aligned} x_k &= A_d^N x_{k-N} + V(N)\overline{u}_{k-1,k-N} \\ \overline{y}_{k-1,k-N} &= U(N)x_{k-N} + T(N)\overline{u}_{k-1,k-N} \end{aligned} \tag{27}$$

where

$$\overline{u}_{k-1,k-N} = \begin{bmatrix} u_{k-1}^T & u_{k-2}^T & \ldots & u_{k-N}^T \end{bmatrix}^T$$

$$\overline{y}_{k-1,k-N} = \begin{bmatrix} y_{k-1}^T & y_{k-2}^T & \cdots & y_{k-N}^T \end{bmatrix}^T$$

$$V(N) = \begin{bmatrix} B_d & A_d B_d & \ldots & A_d^{N-1} B_d \end{bmatrix}$$

$$U(N) = \begin{bmatrix} (CA_d^{N-1})^T & (CA_d)^T & \ldots & C^T \end{bmatrix}^T$$

$$T(N) = \begin{bmatrix} 0 & CB_d & CA_d B_d & \cdots & CA_d^{N-2} B_d \\ 0 & 0 & CB_d & \cdots & CA_d^{N-3} B_d \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 & CB_d \\ 0 & 0 & \cdots & 0 & 0 \end{bmatrix}$$

and $N = \max(\rho_u, \rho_v)$ is a measure of observability, where ρ_u is a minimum value that may be used to create $U(\rho_u)$ full column rank, and ρ_v is the smallest number that can create $V(\rho_v)$ full row rank. Here we observe a lemma on the uniqueness of state reconstruction.

Lemma 1. The recorded inputs and outputs sequences are employed to correctly ascertain the system state for a specific controllable and observable system (20) – (21):

$$x_k = \Theta z_k\,, \tag{28}$$

Where

$$M_u = V(N) - M_y T(N)\,,\ M_y = A_d^N U^+(N)\,,\ \Theta = \begin{bmatrix} M_u & M_y \end{bmatrix},\ z_k = \begin{bmatrix} \overline{u}_{k-1,k-N}^T & \overline{y}_{k-1,k-N}^T \end{bmatrix}^T \in \mathbb{R}^q\,, \text{and } q = N\left[\dim(u) + \dim(y)\right]$$

It is now possible to develop the online output feedback learning method for linear discrete system (20)-(21) based on (25)-(26).

$$x_{k+1} = A_j x_k + B_d \left(K_j x_k + u_k \right) \tag{29}$$

where $A_j = A_d - B_d K_j$. Based on (25) and (29) and using $\overline{K}_j = K_j \Theta$ and $\overline{P}_j = \Theta^T P_j \Theta$, it follows

$$z_{k+1}^T \overline{P}_j z_{k+1} - z_k^T \overline{P}_j z_k = \phi_\kappa^1 \operatorname{vec}\left(\overline{H}_j^1\right) + \phi_\kappa^2 \operatorname{vec}\left(\overline{H}_j^2\right) - \left(y_k^T Q y_k + z_k^T \overline{K}_j^T R \overline{K}_j z_k \right) \tag{30}$$

in which $\overline{H}_j^1 = B_d^T \overline{P}_j B_d$, $\overline{H}_j^2 = B_d^T \overline{P}_j A_d \Theta$, $\phi_\kappa^1 = u_k^T \otimes u_k^T - \left(z_k^T \otimes z_k^T \right)\left(\overline{K}_j^T \otimes \overline{K}_j^T \right)$, $\phi_\kappa^2 = 2\left[\left(z_k^T \otimes z_k^T \right)\left(I_q \otimes \overline{K}_j^T \right) + \left(z_k^T \otimes u_k^T \right) \right]$.

From adaptive theory of control is introduced this presumption of the persistent excitation [13]. Hence, $\overline{K}_{j+1}$ can be calculated as

$$\overline{K}_{j+1} = \left(R + \overline{H}_j^1 \right)^{-1} \overline{H}_j^2 \tag{31}$$

In this case, policy evaluation (30) is utilized to solve $\overline{P}_j$ in a unique way, and policy improvement (31) is implemented for updating control gain $\overline{K}_{j+1}$.

5. RESULTS OF SIMULATIONS

In the following section, we simulate the mobile robot to demonstrate how efficient the output feedback ADP control method is in situations when the states are not measurable and the matrices of the system are unknown. We estimated the best control gain and performance index for the discrete-time system using iterative computations. Additionally, a zero-order holder is used to apply the control strategy for discrete systems to the continuous plant. Employed the sampling time is $h = 0.1\text{s}$.

Robot parameters which are used in simulations are: $M = 22\text{kg}$, $m = 0.41\text{kg}$, $R = 0.105\text{m}$, $D = 0.44\text{m}$, $L = 0.29\text{m}$, $J_y = 0.3384\text{kgm}^2$, $J_z = 0.62\text{kgm}^2$, $J_\omega = 2.39 \cdot 10^{-3}\,\text{kgm}^2$.

Weight matrices Q and R are unit matrices, while for learning purposes, in a period of 4 s, optimal exploration noise in the form of sums of sinusoids was used:

$$e_i(t) = \sum_{j=1}^{6} \sin \omega_{ij}(t) \tag{32}$$

where the optimal frequencies for both inputs are

$$\omega = \begin{bmatrix} -4.65 & 15.44 & 17.90 & 44.51 & 20.92 & -38.06 \\ 39.09 & -16.58 & 19.87 & -30.21 & -46.94 & 24.40 \end{bmatrix} \tag{33}$$

Also, in simulations, it was determined that the state vector's original value is $x_0 = \begin{bmatrix} 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 \end{bmatrix}^T$, while the convergence threshold is $\varepsilon = 10^{-10}$. In this case, the observability index is $N = 6$. The input and output data are gathered from 0.8 to 4 seconds, with the PI initiating at $t = 4\,\text{s}$. Throughout the entire process, the input and output information is gathered online, and the adaptive optimal controller is calculated in an iterative manner. We find estimated optimum values after seven iterations, which are displayed below:

$$\overline{P}_7 = \begin{bmatrix} 1.603 \cdot 10^{2} & 5.348 \cdot 10^{-1} & 2.059 \cdot 10^{1} & 6.339 & -6.122 & 3.622 \cdot 10^{-2} \\ 5.349 \cdot 10^{-1} & 7.567 \cdot 10^{-2} & 1.231 \cdot 10^{-1} & 1.373 \cdot 10^{-1} & 3.454 \cdot 10^{-1} & 4.666 \cdot 10^{-4} \\ 2.059 \cdot 10^{1} & 1.232 \cdot 10^{-1} & 3.169 \cdot 10^{1} & -1.104 \cdot 10^{-1} & 9.615 & 3.173 \cdot 10^{-3} \\ 6.339 & 1.373 \cdot 10^{-1} & -1.105 \cdot 10^{-1} & 4.605 & -6.785 \cdot 10^{-1} & -2.422 \cdot 10^{-3} \\ -6.121 & 3.453 \cdot 10^{-1} & 9.625 & -6.785 \cdot 10^{-1} & 2.441 \cdot 10^{1} & 2.218 \cdot 10^{-3} \\ 3.622 \cdot 10^{-2} & 4.666 \cdot 10^{-4} & 3.173 \cdot 10^{-3} & -2.424 \cdot 10^{-3} & 2.221 \cdot 10^{-3} & 5.734 \cdot 10^{-3} \end{bmatrix} \tag{34}$$

$$\overline{K}_7 = \begin{bmatrix} -3.413 \cdot 10^{-1} & -6.281 \cdot 10^{-2} & -9.496 \cdot 10^{-2} & 4.114 \cdot 10^{-1} & -3.772 \cdot 10^{-1} & -9.278 \cdot 10^{-4} \\ 1.975 & 1.176 \cdot 10^{-1} & 1.655 \cdot 10^{-1} & 2.696 \cdot 10^{-1} & 9.821 \cdot 10^{-2} & 9.339 \cdot 10^{-4} \end{bmatrix} \tag{35}$$

The inputs and outputs of the robot controlled by the ADP-based controller are shown in Fig. 2.

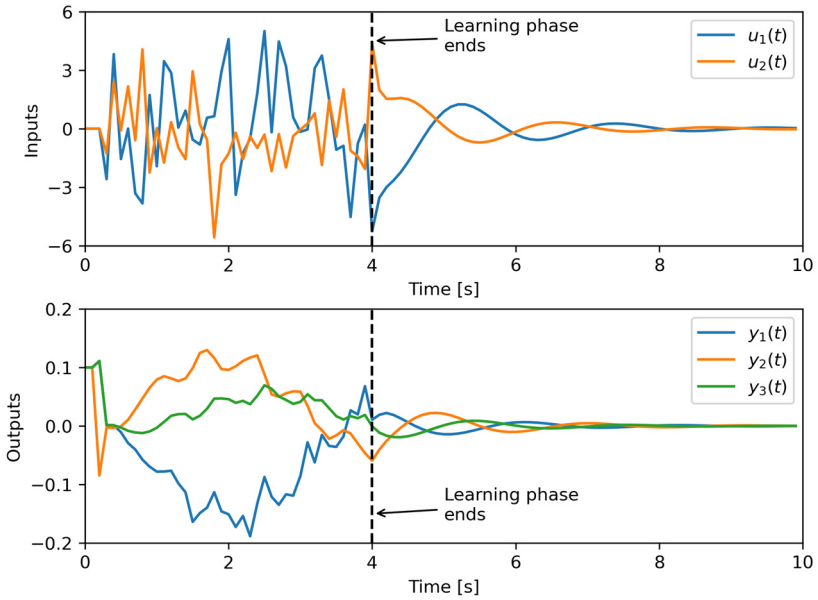

Figure 2: Input and output signals of the robot

Figure 3 shows the change in the state vector norm over time, while Figure 4 shows the state trajectories of the mobile robot. From these figures it can be seen that the states converge to the equilibrium state quickly after the controller is updated.

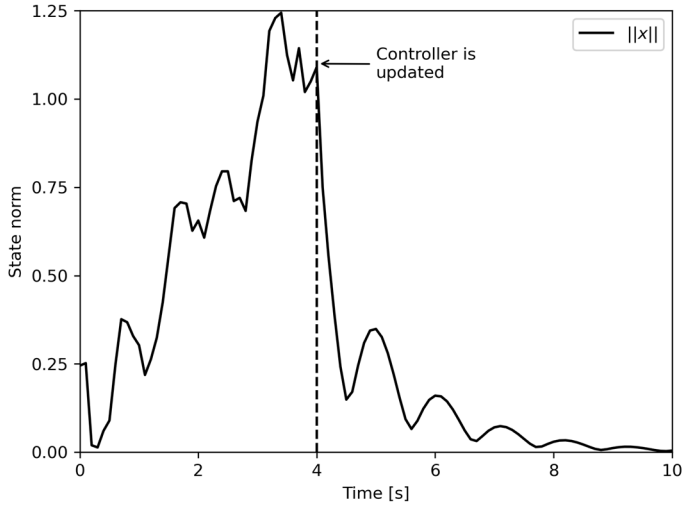

Figure 3: The norm of the robot's state vector

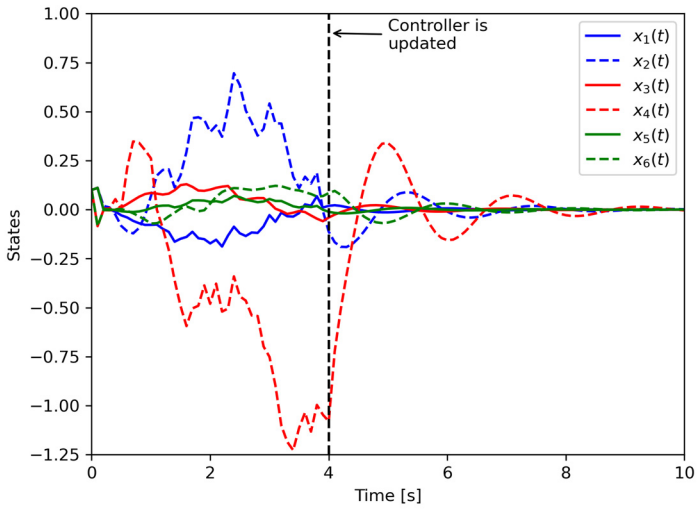

Figure 4: Trajectory of states

Fig. 5 shows the convergence of the approximate values of the matrices $\bar{P}_j$ and $\bar{K}_j$ in direction of the optimal values $\bar{P}^*$ and $\bar{K}^*$, respectively.

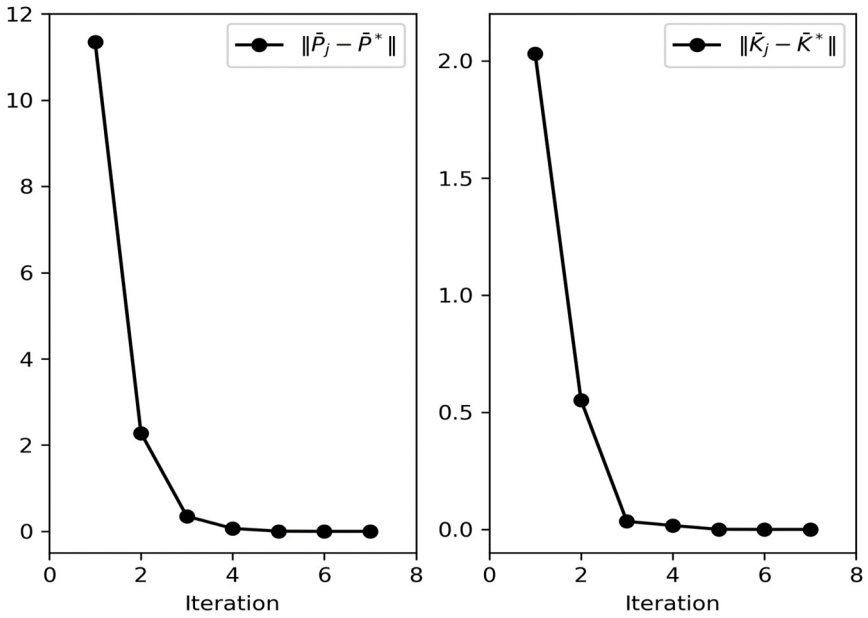

Figure 5: Convergence of the approximated matrices $\bar{P}_j$ and $\bar{K}_j$

The distinction between the approximation and optimal value functions is seen in Figure 6. The exiguous difference implies a good match between the approximate and optimal value functions. Figure 7 shows the difference between the optimal and approximate control laws. Based on Figure 7, the approximate and optimal control laws practically agree.

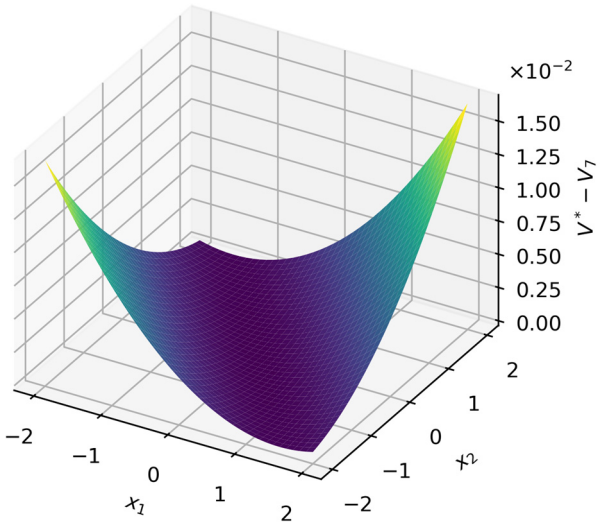

Figure 6: Difference between optimal and approximate value function

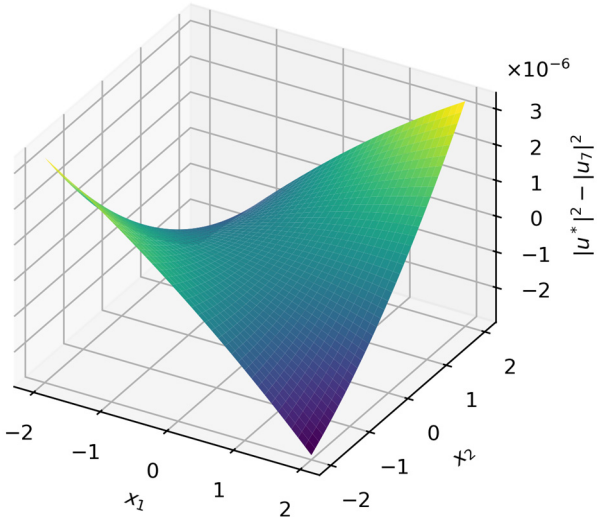

Figure 7: Difference between the optimal and approximate control law

6. CONCLUSIONS

This paper examines the design of an optimal controller based on adaptive dynamic programming (ADP) for a two-wheeled, self-balancing mobile robot with entirely unidentified dynamics. The fundamental advantage of the provided control mechanism is that it avoids knowledge of the complete system's dynamics, which is critical in real-world situations. An efficient strategy in these situations is the adaptive optimum control technique with sampled data, which is based on a discrete model and output feedback. The online control strategy that is being given learns the optimal control gain only by using measured input and output data. When it comes to control methods that involve direct state measurement and several sensors, the reconstruction of states is quite useful. The simulation outcomes indicate that the proposed control approach is valid and effective.

ACKNOWLEDGEMENTS

This study was supported by the Serbian Ministry of Science, Technological Development and Innovation from the grant 451-03-65/2024-03/200108.

REFERENCES

[1] R.S. Sutton and A.G. Barto, "Reinforcement learning: An introduction", MIT Press, Cambridge, Massachusetts (USA), (2020)

[2] B. Tao and Z. P. Jiang, "Value iteration and adaptive dynamic programming for data-driven adaptive optimal control design", Automatica, Vol. 71, pp. 348-360, https://doi.org/10.1016/j.automatica.2016.05.003, (2016)

[3] M. Roozegar, M. J. Mahjoob, and M. Jahromi, "Optimal motion planning and control of a nonholonomic spherical robot using dynamic programming approach: simulation and experimental results", Mechatronics, Vol. 39, pp. 174-184, https://doi.org/10.1016/j.mechatronics.2016.05.002, (2016)

[4] J. H. Zhu, X. S. Ge, and M. Z. Wang, "Adaptive dynamic programming method for attitude control of three-axis spacecraft", Journal of Beijing University of information science and technology: natural science edition, (2018)

[5] F. L. Lewis and K. G. Vamvoudakis, "Reinforcement learning for partially observable dynamic processes: adaptive dynamic programming using measured output data", IEEE Trans. Syst. Man Cybern. B, Cybern,Vol. 41(1), pp. 14–25, https://doi.org/10.1109/TSMCB.2010.2043839, (2011)

[6] M. T. Rodriguez and S. P. Banks, "Linear, time-varying approximations to nonlinear dynamical systems", Berlin Springer (Germany), (2010)

[7] M. Mynuddin and W. Gao, "Distributed predictive cruise control based on reinforcement learning and validation on microscopic traffic simulation", IET Intelligent Transportation Systems, Vol. 14(5), pp. 270-277, https://doi.org/10.1049/iet-its.2019.0404, (2020)

[8] A. Cavallo, G. de Maria, C. Natale, and S. Pirozzi, "Slipping detection and avoidance based on Kalman filter", Mechatronics Vol. 24(5), pp. 489–499, https://doi.org/10.1016/j.mechatronics.2014.05.006, (2014)

[9] W. Gao, M. Huang, Z. P. Jiang, and T. Chai, "Sampled-data-based adaptive optimal output-feedback control of a 2-degree-of-freedom helicopter" IET Control Theory & Applications, Vol. 10, pp. 1440-1447, https://doi.org/10.1049/iet-cta.2015.0977, (2016)

[10] L. Guo, S. A. A. Rizvi, and Z. Lin. "Optimal control of a two-wheeled self-balancing robot by reinforcement learning", International Journal of Robust and Nonlinear Control, Vol. 31(6), pp. 1885–1904, https://doi.org/10.1002/rnc.5058, (2021)

[11] T. Chen and B. A. Francis, "Optimal sampled-data control systems", Springer-Verlag, New York (USA), (1995)

[12] G. Hewer, "An iterative technique for the computation of the steady state gains for the discrete optimal regulator", IEEE Transactions on Automatic Control, Vol. 16(4), pp. 382–384, https://doi.org/10.1109/TAC.1971.1099755, (1971)

[13] K. J. Astrom and B. Wittenmark, "Adaptive control", Dover Publishing Inc., Mineola, New York (USA), (1994)