Observer based state feedback controller design of a DC servo motor using identified motor model: an experimental study

Sobia Shafi
Dept. of Electrical Engineering
IUST
Awantipora, India
sobia.shafi19@gmail.com

Peerzada S. Hamid
Dept. of Electronics and Communication Engineering
IUST
Awantipora, India
pzshoaib@gmail.com

Shahkar A. Nahvi
Dept. of Electrical Engineering
IUST
Awantipora, India
shahkar@iust.ac.in

Majid H. Koul
Dept. of Mechanical Engineering
IUST
Awantipora, India
majid.koul@islamicuniversity.edu.in

Mohammad A. Bazaz
Dept. of Electrical Engineering
NIT
Srinagar, India
abid@nitsri.ac.

2023 International Conference on Power, Instrumentation, Energy and Control (PIECON) | 979-8-3503-9976-9/23/$31.00 ©2023 IEEE | DOI: 10.1109/PIECON56912.2023.10085764

***Abstract*—Full state feedback is a commonly used technique in control systems. However, full state feedback control implementation requires knowledge of the entire state vector of the system which is not always possible. Thus, observers are used to estimate the immeasurable states which are then used for state feedback control. This work presents an experimental procedure to control the angular position of a DC servo motor using state feedback control law. The process involves system identification, followed by observer and controller design using separation principle.**

***Index Terms*—Luenberger Observer, State feedback control, System identification**

I. Introduction

A DC servo motor is one of the most frequently used actuators in mechatronic systems [1]. It can directly provide rotatory motion and when coupled with other elements, it also provides translational motion. In most of the applications, precise positioning of the motor shaft is required [2] which is achieved using various control strategies such as classical PID control, full state feedback control, linear quadratic regulator control [3] and model predictive control [4].

Full state feedback control, also know as pole placement method uses the state vector to place the closed-loop poles of a plant in specified locations in the s-plane [5], [6]. Full state feedback control of a DC servo motor is based on the assumption that the entire state vector is measurable and hence available for use. However, in most of the situations measuring all the states is either impossible or prohibitively expensive and thus only few states of the system are available [7]. In these cases a suitable approximation or estimate of a state vector is used. The state vector is estimated from input and output measurements of the system using a state estimator or an observer [8]. An observer is a simulated mathematical model of the actual physical system which is to be controlled. It provides an estimate of the states of the system that are not measurable. Observers have been used for parameter estimation, for detecting novices in the motion control applications [9], designing robust controllers for trajectory tracking [10], and sensor fault detection [11]. Various types of observers include Luenberger observer, reduced order observer and Kalman filters. This work demonstrates the design and implementation of a Luenberger observer to estimate the states of a DC motor. The estimated states are used to design a full state feedback controller to control the angular position of the DC servo motor.

The rest of the paper is organized as follows: Section II describes the hardware setup. Section III describes the system governing equations and System Identification. Section IV describes the observer design based on identified model and its implementation both in simulation and actual set-up. Section V describes the full state controller design and its implementation followed by Section VI and Section VII which discuss the results and conclusion respectively.

II. Hardware Set-up

The experimentation was carried out using an AndyMark-4252, 12-volt, brushless DC servo motor. The motor is equipped with an inbuilt optical encoder to keep track of the motor shaft position. It provides 5376 counts per revolution at the output shaft. A cytron (MD10C) DC motor driver was used to drive the motor. The motor was controlled using an Arduino Uno board. The entire setup was powered using a

The study was undertaken with financial support from Jammu and Kashmir Science and Technology Innovation Council (JKSTIC). JKSTIC is not responsible for any results/interpretations expressed in the study.

12-volt DC power supply. Fig. 1 shows the actual hardware setup used for experimentation.

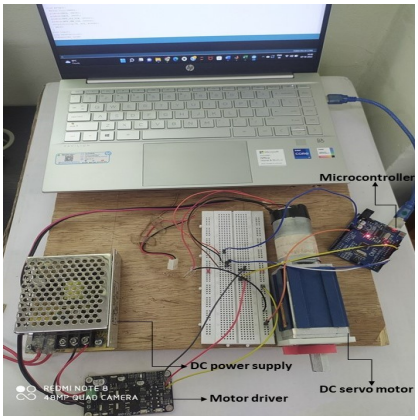

Fig. 1. Hardware setup used for the experimentation

III. DC SERVO MOTOR MODELLING AND SYSTEM IDENTIFICATION

The first step in the process of implementing the full state feedback control is to identify the model of the system, DC servo motor in our case. The identified model is used to design an observer which would provide an estimate of all the states of the system. Since the system equations governing the DC motor model are already known, we used Grey Box modelling for parameter estimation [12] [13]. System equations and parameter estimation is described below.

A. DC Servo Motor System Equations

In this section we describe the governing equations of DC servo motor. Consider an armature-controlled DC servo motor whose equivalent circuit diagram is shown in Fig. 2. The transfer function of the motor, $G_{\text{position}}(s)$, where the input is the applied armature voltage, $E_{\text{a}}(s)$, and the output is the angular position of the motor, $\theta(s)$, is given by [14]

$$\begin{aligned} G_{\text{position}}(s) &= \frac{\theta(s)}{E_{\text{a}}(s)} \\ &= \frac{K_{\text{t}}}{s((J_{\text{m}}+B_{\text{m}})(R_{\text{a}}+sL_{\text{a}})+K_{\text{t}}K_{\text{b}})} \end{aligned} \tag{1}$$

where K_{t} is the motor torque constant, J_{m} is the equivalent inertia at the armature, B_{m} is the equivalent viscous damping coefficient at the armature, R_{a} is the armature resistance, L_{a} is the armature inductance and K_{b} is the back electromotive force constant.

Usually for a DC servo motor, the armature inductance L_{a}, is small compared to armature resistance R_{a}, so after simplification, (1) becomes

$$\frac{\theta(s)}{E_{\text{a}}(s)} = \frac{K_{\text{m}}}{s(\tau_{\text{m}}s+1)} \tag{2}$$

where $K_{\text{m}} = \frac{K_{\text{t}}}{B_0 R_{\text{a}}}$, is the motor gain constant, $\tau_{\text{m}} = \frac{J_{\text{m}}}{B_0}$, is the motor time constant and $B_0 = B_{\text{m}} + \frac{(K_{\text{t}}K_{\text{b}})}{R_{\text{a}}}$

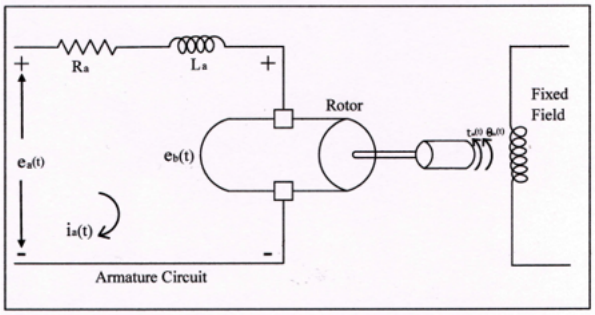

Fig. 2. DC servo motor equivalent circuit diagram

$G_{\text{speed}}(s)$ (transfer function relating armature voltage as input and motor angular velocity as output) is obtained as

$$G_{\text{speed}}(s) = \frac{s\theta(s)}{E_{\text{a}}(s)} = \frac{\dot{\theta}(s)}{E_{\text{a}}(s)} = \frac{K_{\text{m}}}{(\tau_{\text{m}}s+1)} \tag{3}$$

Equation (2) can be written in time domain as:

$$\ddot{\theta}(t) = \frac{K_{\text{m}}}{\tau_{\text{m}}} e_{\text{a}}(t) - \frac{1}{\tau_{\text{m}}}\dot{\theta}(t) \tag{4}$$

We convert the second order Ordinary Differential Equation (ODE) given by (4) to standard state space form by employing the following substitutions for state variables:

$$x_1(t) = \theta(t);\ x_2(t) = \dot{\theta}(t)$$

This gives the state space form a follows

$$\dot{x}(t) = Ax(t) + Bu(t);\ y(t) = Cx(t) \tag{5}$$

where states $x(t) \in \mathbb{R}^2$, $u(t) \in \mathbb{R}$ and is the input to the system, which in this case is the applied armature voltage $e_{\text{a}}(t)$, and $y(t) \in \mathbb{R}$ is the output of the system. The state matrix A, the input matrix B and the output matrix C, are hence obtained as:

$$A = \begin{bmatrix} 0 & 1 \\ 0 & -1/\tau_{\text{m}} \end{bmatrix},\ B = \begin{bmatrix} 0 \\ K_{\text{m}}/\tau_{\text{m}} \end{bmatrix},\ C = \begin{bmatrix} 1 & 0 \end{bmatrix}$$

B. DC Servo Motor System Identification

The next step is to estimate the constants, τ_{m} and K_{m} , in the above governing equations. This is done using two separate approaches.

In the first approach we used step-speed response of the motor. A step input is given to the DC servo motor and the output response of the motor in terms of angular position of the motor shaft is recorded using the inbuilt optical encoder. The angular velocity of the motor shaft is calculated from angular position readings and the velocity-time graph is plotted. The same is shown in Fig. 3. From the graph, the steady state value or motor gain constant, K_{m}, and the motor time constant, τ_{m} are calculated which are then used to determine the DC servo motor model in the form of transfer function and state space representation.

It can be readily seen from Fig. 3, that motor speed reaches a steady state value of 1.695 radians per second and hence (motor gain constant), $K_{\mathrm{m}} = 1.695$ radians per second. The time constant,τ_{m} which is measured by calculating the time for the velocity to reach 63% of its steady state value is 0.024 sec and hence $\tau_{\mathrm{m}} = 0.024$ seconds.

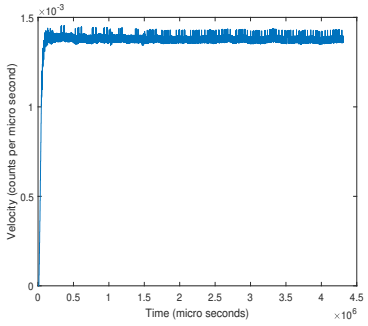

Fig. 3. Step response of the DC servo motor

Thus, the open loop transfer function of the DC servo motor given by (2 and 3) can be re-written as

$$G_{\text{position}}(s) = \frac{\theta(s)}{E_{\mathrm{a}}(s)} = \frac{1.695}{s(0.024s+1)} \tag{6}$$

$$G_{\text{speed}}(s) = \frac{\dot{\theta}(s)}{E_{\mathrm{a}}(s)} = \frac{1.695}{0.024s+1} \tag{7}$$

Also the matrices A and B used in state space representation of the system given by (5) are thus identified as:

$$A = \begin{bmatrix} 0 & 1 \\ 0 & -41.67 \end{bmatrix}, \; B = \begin{bmatrix} 0 \\ 70.625 \end{bmatrix}$$

This process is repeated for different values of step inputs and the transfer function is found to be same.

An alternative approach to estimate the above constants is by using the System Identification toolbox from MATLAB. In this method, DC servo motor is excited using a sine input with variable amplitude and frequency. Angular position of motor shaft is again recorded using the inbuilt optical encoder. The angular velocity of the motor shaft is then calculated from the angular position data. This time domain input and output data (armature voltage and shaft angular velocity respectively) is imported in the System Identification toolbox. This data is broken into two sets in the ratio of 80:20. The first set is used to identify the model (transfer function in the form of (3)) and the second set is used to validate the identified model. The transfer function thus estimated is identical to the one found using the first approach. The identified model is now used to design the observer which is discussed in the next section.

IV. Observer Design and Implementation

An observer is a simulated mathematical model of the actual physical system which is to be estimated/controlled. It provides an estimate of the states of the system that are either impossible or prohibitively expensive to measure [15]. In control design, the estimated states are then combined with the measured states to implement state feedback control. The only and sufficient condition for an observer to be successfully implemented is that the system should be observable. The Linear time-invariant (LTI) system in (5) is said to be observable if the knowledge of $u(t)$ and $y(t)$ over some finite time interval $0 < t \leq t_{\mathrm{k}}$ is enough to uniquely determine x_0 provided A, B, C and D are known [16]. Consequently, a system is said to be observable if and only if the observability matrix, $\mathcal{O}$ is full rank.
where

$$\mathcal{O} = \begin{bmatrix} C \\ CA \\ CA^2 \\ \vdots \\ CA^{n-1} \end{bmatrix}$$

Various types of observers cited in literature include Luenberger observer, reduced order observer and Kalman filters. In this work, we have implemented a Luenberger Observer which is discussed next.

A. Luenberger Observer

Luenberger Observer (LO) is a linear state observer used to provide an estimate of all the states of the system. It has its design based on eigen value assignment or pole placement. The block diagram of Luenberger observer is shown in Fig. 4. For the system in (5), the Luenberger observer is mathematically represented as

$$\begin{aligned} \dot{\hat{x}}(t) &= A\hat{x}(t) + Bu(t) + L(y(t) - \hat{y}(t)) \\ &= (A - LC)\hat{x}(t) + Bu(t) + Ly(t) \end{aligned} \tag{8}$$

$$\hat{y} = C\hat{x}(t) \tag{9}$$

where $\hat{x}(t)$ is the estimate of the actual state $x(t)$ and L is a 2×1 constant observer gain matrix.

Representing the error between the actual and estimated states by $\tilde{x}(t)$, where $\tilde{x}(t) = x(t) - \hat{x}(t)$ and using (5) & (8, 9) to write error dynamics, we obtain:

$$\dot{\tilde{x}}(t) = (A - LC)\tilde{x}(t) \tag{10}$$

Mathematically, the error, $\tilde{x}(t)$ tends to 0 as time, t tends to infinity. The speed of the convergence of error can be adjusted by appropriately selecting the observer gain values, L while at the same time ensuring that the eigen values of matrix, $A - LC$, are negative.

The observer implementation thus needs to solve the differential equation (8). The same was implemented using 4^{th} order Runge-Kutta (RK) method which is discussed next.

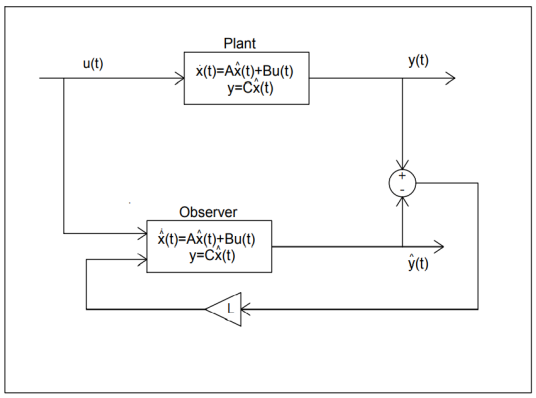

Fig. 4. Block diagram of a Luenberger Observer

B. Runge-Kutta method

Runge-Kutta (RK) method is a widely used numerical method to solve differential equations. The most widely known member of the Runge–Kutta family is generally referred to as '$RK4'$ or the 'classic 4^{th} Runge–Kutta method'. This classic 4^{th} order RK method is used to solve the observer dynamical equations (8, 9) to obtain the solution $\hat{x}$, which is the estimated state for some randomly assumed initial condition, $\hat{x}(0)$. A brief overview of the RK method is given below.

Assume the following initial value problem

$$\frac{dx}{dt} = f(x,t);\ x(t_0) = x_0 \tag{11}$$

Assuming a step-size, $h > 0$ we define the update equation as:

$$x_{\text{n+1}} = x_{\text{n}} + h\frac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4) \tag{12}$$

where $k1$, $k2$, $k3$ and $k4$ are given by

$$k_1 = f(t_{\text{n}}, x_{\text{n}});\ k_2 = f(t_{\text{n}} + \frac{h}{2}, x_{\text{n}} + h\frac{k_1}{2})$$

$$k_3 = hf(t_{\text{n}} + \frac{h}{2}, x_{\text{n}} + h\frac{k_2}{2});\ k_4 = hf(t_{\text{n}} + h, x_{\text{n}} + hk_3)$$

The pseudocode for implementing the above equations is given in Algorithm 1.

Algorithm 1 Runge Kutta 4th order method

In this code a function ODE is used which has the current time (t), the state vector ($y \in R^{1\text{xn}}$) and the array that needs to be updated ($Yp \in R^{1\text{xn}}$), as its arguments. This code requires the step size (dt), a termination condition (T), initial state of ODEs ($Ycurrent \in R^{1\text{xn}}$) and number of state variables (n) from the user. Initialize the current time (t), a variable for temporarily storing the ODE function output ($Ytemp \in \mathbb{R}^{1\text{xn}}$) and RK intermediate points ($k1 \in R^{1\text{xn}}, k2 \in R^{1\text{xn}}, k3 \in R^{1\text{xn}}, k4 \in R^{1\text{xn}}$)
Call ODE (t,Ycurrent,k1)
return k1
for $i \leftarrow 1$ to n **do**
    $Ytemp(i) \leftarrow Ycurrent(i) + k1(i) * dt * 0.5$
    Call ODE (t+dt*0.5,Ytemp,k2)
    return k2
    for $i \leftarrow 1$ to n **do**
        $Ytemp(i) \leftarrow Ycurrent(i) + k2(i) * dt * 0.5$
        Call ODE (t+dt*0.5,Ytemp,k3)
        return k3
        for $i \leftarrow 1$ to n **do**
            $Ytemp(i) \leftarrow Ycurrent(i) + k3(i) * dt$
            Call ODE (t+dt,Ytemp,k4)
            return k4
            for $i \leftarrow 1$ to n **do**
                $Ytemp(i) \leftarrow Ycurrent(i) + dt/6 * (k1(i) + 2 * k2(i) + k3(i) + k4(i))$
                t+dt
                if $t \leq T$ **then** ▷ T is the final time upto which the value of ODE is to be calculated
                    go to 2
                else
                    go to 24
                end if
                end
                function ODE((t,Y,Yp))
                    $Yp(1) \leftarrow -l1 * Y(1) + Y(2) + l1 * y$▷ From Eq.3.1
                    $Yp(2) \leftarrow -l2 * Y(1) - a22 * Y(2) + b2 * u + l2 * y$
                end function=0

V. Controller Design by Separation Principle for Servo Control

The last step is to design the controller which is designed using the separation principle. The measured physical states along with the states estimated by the observer are fed back to the system in order to implement feedback control. For the system in (5), the control law for the output to reach a desired steady state value (r), is given as

$$u = -Kx(t) + Nr \tag{13}$$

where u is the control input and $K = [K_1, K_2]$ is controller gain matrix, N is the scaling term given by $N^{-1} = -C(A - BK)^{-1}B$ and r is the constant reference input.

Using separation principle, controller and observer are designed separately and are together used to implement the full state feedback for position control of a DC servo motor [17] [18]. A general rule of thumb is to choose the observer poles to be two to five times faster than the system response, to ensure faster decay of estimation error, causing the closed loop poles from the full state feedback to dominate the total response[19].

VI. Results

This section discusses the results obtained during the implementation of the proposed method.

The condition for the plant to be observable (with voltage as input and angular position as output) was first checked.

Since the plant was found to be observable, the LO was designed based on the identified model, to estimate all the states of the system using (8 and 9) and the observer gains were heuristically fixed at $l_1 = 240$ and $l_2 = 19200$. The error in estimation of the velocity is plotted in Fig. 6. It is clearly seen that starting with an error in estimation of the initial state $x(0)$, the error quickly decays to zero as was expected by design.

The designed observer was first simulated in MATLAB and Simulink. The estimated position was compared with the measured position and the estimated velocity was compared with the velocity calculated by differentiating the measured position. The same is plotted in Fig. 7 and it is clearly seen that the estimated position and velocity are in sync with the measured position and calculated velocity respectively. Next the designed observer was implemented in the Arduino Uno to estimate the states of the dc servo motor. The estimated position was compared with the measured position using the inbuilt optical encoder and are plotted in Fig. 8. The measured position was used to calculate the velocity and same was compared with the estimated velocity. Both are plotted in Fig. 9. It is seen that both, estimated position and estimated velocity are in sync with the measured position and calculated velocity respectively.

The designed observer was used to implement full state feedback control of angular position for the DC servo motor. The reference input was chosen to be a step function. The settling time for output response was selected to be $2s$ with no overshoot and no steady state error. The controller gain matrix $K = [K_1, K_2]$ was chosen such that the response specifications are met. The same was implemented on the hardware set-up. The output response was measured and same is plotted in Fig. 10. It is seen that the response obtained from the hardware set-up matches closely with the response specifications for which the observer based full state feedback controller was designed.

VII. Conclusion

In this paper, a full state feedback control technique is presented for trajectory tracking of the angular position of a DC servo motor. The motor model was first identified using the step response of the motor from which the motor time and gain constants were calculated to get the motor transfer function. A state space representation of the motor was derived from the obtained transfer function, based on which a Luenberger Obserever was designed to estimate the system states. The observed system states were compared with the measured states and both the estimated and measured states were found to converge in simulation as well as hardware experimentation. The estimated states were then used to implement full state feedback controller for angular position control of a DC servo motor to reach the given set point and the desired system response which satisfies the given time specifications was achieved.

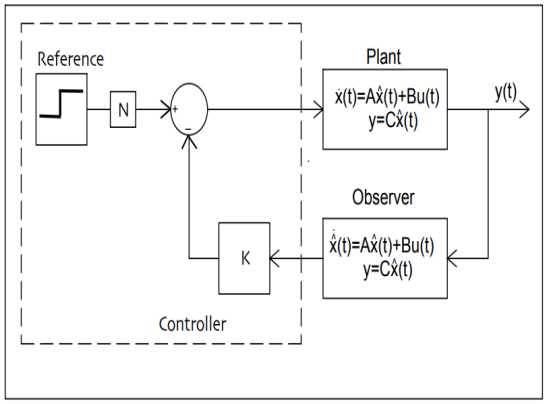

Fig. 5. Block diagram of servo control using full state feedback

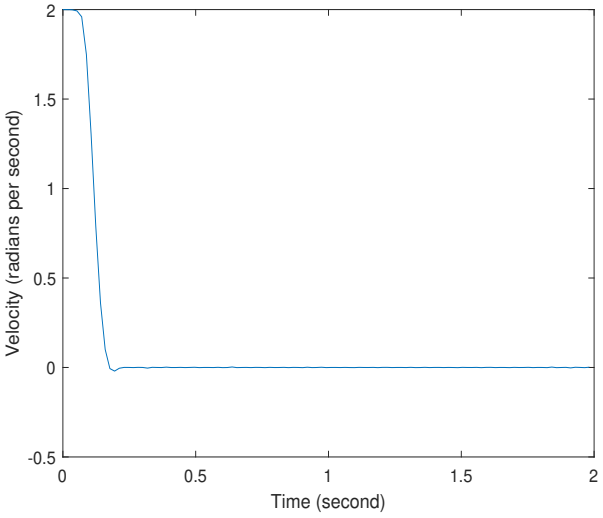

Fig. 6. Error dynamics of estimated velocity with different initial conditions

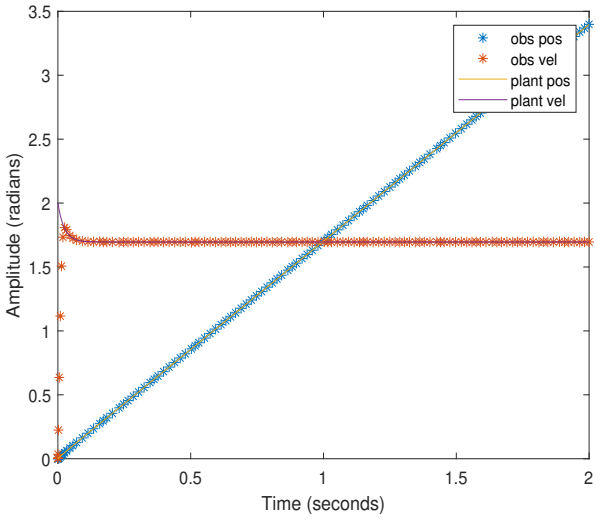

Fig. 7. Simulation of estimation of states using a Luenberger Observer

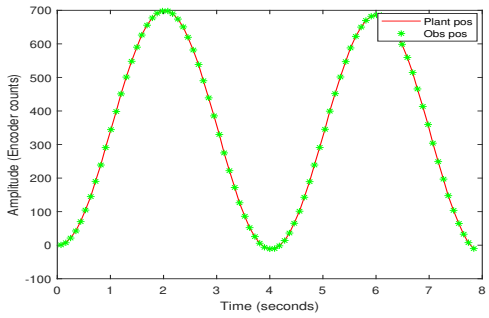

Fig. 8. Estimation of position using Luenberger Observer with sine input

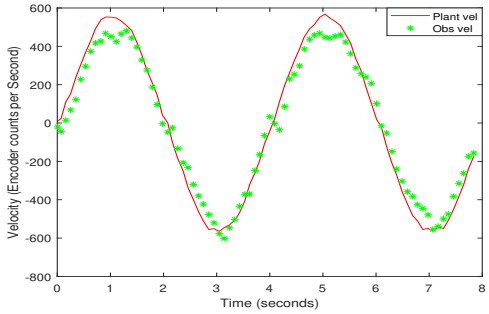

Fig. 9. Estimation of velocity using Luenberger Observer with sine input

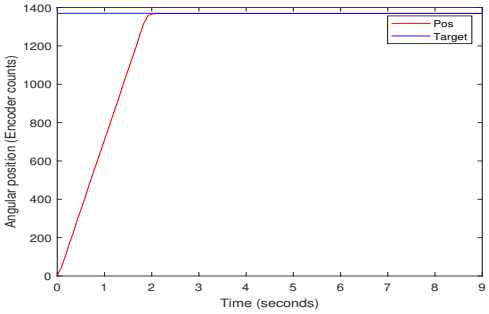

Fig. 10. Full state feedback servo control response for a step input, implemented in hardware

REFERENCES

[1] H. Zhou, "Dc servo motor pid control in mobile robots with embedded dsp," in *2008 International Conference on Intelligent Computation Technology and Automation (ICICTA)*, vol. 1, pp. 332–336, IEEE, 2008.

[2] A. Chowdhury and D. Debnath, "Performance comparison between pid controller and state-feedback controller with integral action in position control of dc motor," in *Applied mechanics and materials*, vol. 367, pp. 188–193, Trans Tech Publ, 2013.

[3] R. M. A.-M. Hummadi, "Simulation of optimal speed control for a dc motor using linear quadratic regulator (lqr)," *Journal of Engineering*, vol. 18, no. 3, 2012.

[4] S. Sahoo, B. Subudhi, and G. Panda, "Optimal speed control of dc motor using linear quadratic regulator and model predictive control," in *2015 international conference on energy, power and environment: towards sustainable growth (ICEPE)*, pp. 1–5, IEEE, 2015.

[5] M. S. Fadali and A. Visioli, *Digital control engineering: analysis and design*. Academic Press, 2012.

[6] E. D. Sontag, *Mathematical control theory: deterministic finite dimensional systems*, vol. 6. Springer Science & Business Media, 2013.

[7] D. Luenberger, "An introduction to observers," *IEEE Transactions on automatic control*, vol. 16, no. 6, pp. 596–602, 1971.

[8] O. Moseler and R. Isermann, "Application of model-based fault detection to a brushless dc motor," *IEEE Transactions on industrial electronics*, vol. 47, no. 5, pp. 1015–1020, 2000.

[9] W. Schmitendorf and C. Hollot, "Simultaneous stabilization via linear state feedback control," *IEEE transactions on automatic control*, vol. 34, no. 9, pp. 1001–1005, 1989.

[10] B. M. Pillai and J. Suthakorn, "Motion control applications: observer based dc motor parameters estimation for novices," *Int. J. Power Electron. Drive Syst*, vol. 10, no. 1, pp. 195–210, 2019.

[11] C. Aguilar-Ibañez, J. Mendoza-Mendoza, J. Davila, M. S. Suarez-Castanon, R. Garrido M, *et al.*, "A robust controller for trajectory tracking of a dc motor pendulum system," *International Journal of Control, Automation and Systems*, vol. 15, no. 4, pp. 1632–1640, 2017.

[12] A. Alkaya and I. Eker, "Luenberger observer-based sensor fault detection: online application to dc motor," *Turkish Journal of Electrical Engineering and Computer Sciences*, vol. 22, no. 2, pp. 363–370, 2014.

[13] N. R. Kristensen, H. Madsen, and S. B. Jørgensen, "Parameter estimation in stochastic grey-box models," *Automatica*, vol. 40, no. 2, pp. 225–237, 2004.

[14] N. S. Nise, *Nise's Control Systems Engineering*. Wiley, 2018.

[15] B. Sohlberg, *Supervision and control for industrial processes: using grey box models, predictive control and fault detection methods*. Springer Science & Business Media, 2012.

[16] G. Ellis, *Observers in control systems: a practical guide*. Elsevier, 2002.

[17] M. Gopal, *Digital cont & state var met*. Tata McGraw-Hill Education, 2012.

[18] M. N. Kamarudin, S. M. Rozali, and A. R. Husain, "Observer-based output feedback control with linear quadratic performance," *Procedia Engineering*, vol. 53, pp. 233–240, 2013.

[19] "State estimation." https://ecal.berkeley.edu/files/ce295/CH02-StateEstimation.pdf. Accessed: 2023-01-14.