SYSTEM IDENTIFICATION OF A HOBBY DC MOTOR USING A LOW COST ACQUISITION SETUP

Igor F. Okuyama*, Marcos R. O. A. Máximo*, Samuel C. Pinto*

*Praça Marechal Eduardo Gomes, 50, Vila das Acácias, 12228-900
Instituto Tecnológico de Aeronáutica
São José dos Campos, São Paulo, Brasil

Emails: franzoni315@gmail.com, maximo.marcos@gmail.com, sacepi@gmail.com

Abstract— Hobby DC motors manufacturers typically do not provide the required parameters values to model the motor dynamic response. However, knowing the model accurately is very important for control, state estimation and simulation. The proposed method uses a low cost acquisition setup and uses a frequency domain analysis based on the Bode diagram technique. To improve the identification accuracy, we also considered issues such as the encoder quantization, the effect introduced by the input zero-order hold and some nonlinearities of the DC motor.

Keywords— System Identification, Motor Modeling, Control, Robotics.

1 Introduction

ITAndroids is a robotics group from Instituto Tecnológico de Aeronáutica (ITA). In 2013 we started to develop the IEEE Very Small Size League (VSS) category, a soccer game consisted of 3 autonomous differential-drive robots per team. The players have a size restriction of 7.5 cm per side and have identification color on its top side. A camera placed above the field sends a video to a computer, that has a vision processing algorithm. This algorithm is able to estimate the position and orientation of each robot and a game strategy program can use this to create commands to the players. VSS rules can be found at (7th Latin American IEEE Robotics Competition, 2008).

To develop an efficient strategy, it is important to test a wide range of different algorithms and this is easier to do through simulations than dealing with real robots. Nevertheless, to simulate the robot it is important to understand its dynamic equations, i.e., we need to know the system. Besides, if we want to design, for example, a position controller for a differential robot using Control Theory, we also need a detailed model. (Dušek et al., 2011) and (Guerra et al., 2004) show how to do this modeling and we need values of many parameters like moment of inertia, lengths and DC motor constants like viscous friction constant, electromotive force constant and moment of inertia of the rotor.

Unfortunately, if hobby DC motors are used, it is probable that it will not have a good documentation and many parameters might be missing. This article will focus on how to overcome this problem. We propose a method based on frequency domain analysis and Bode diagram to characterize the motor, by using the same hardware used in the robot, i.e., using a processor and a magnetic encoder. Thus, we did not need to add any electronics to our robot.

The work is organized this way: in Sec. 2, we present the acquisition system (the hardware used), and we analyze the encoder error and the input zero-order hold effect. We also present in this section the linear motor model used and some nonlinearities that could affect the system. Section 3 shows the experimental method used to identify the system and how we could validate the model obtained. Section 4 concludes the article.

2 The Acquisition System

2.1 The Hardware

Our acquisition hardware is composed by a processing unit, a 16 MHz microcontroller (Arduino Nano); an H-Bridge (Pololu’s TBF6612FNG); a Pololu micro-motor (the one we want to characterize) with 51.45:1 gearbox and with extended shaft; a 12 CPR (counts per revolution) Pololu magnetic quadrature encoder and a 7.4 V Lipo battery. The final encoder resolution is \( N_p = 51.45 \times 12 = 617.4 \) CPR. Specification of the motor and the encoder can be found respectively at (Pololu, 2015b) and (Pololu, 2015a). Arduino Nano and battery can easily be found at any electronics or robotics store. As ITAndroids’ VSS robot (Fig. 1) already had all these components, we used it as the acquisition system. Thus, we did not have to add any electronics to the robot.

Using this setup, we can excite the motor with a PWM signal generated by the microcontroller and measure the speed or position using the quadrature encoder. Figure 2 shows the block diagram of our setup.

2.2 Encoder Error Analysis

According to (Petrella et al., 2007), there are two common methods for measuring velocity with encoders: frequency measurement and period measurement. Figure 2 shows the block diagram of our setup.
time and counting how many encoder pulses happened in this window. The second is accomplished by measuring the time between two pulses. Nevertheless, the mathematical analysis of discrete-time dynamical systems with fixed sampling rate is well understood in the Digital Control Literature, therefore we choose the frequency measurement, described by:

\[ \omega = \frac{d\theta}{dt} \approx \frac{\Delta \theta}{T_s} = \frac{2\pi \times \Delta N \text{ rad}}{N_p T_s} s \]  

(1)

where \( \Delta N \) is the number of pulses which happened inside the time-window \( T_s \) in seconds, and \( N_p \) is the encoder resolution. As the time-window is not synchronized with the pulses, we can miscount the number of pulses by one. This leads to a quantization error \( \Delta \omega \):

\[ \Delta \omega = \frac{2\pi \text{ rad}}{N_p T_s} s \approx \frac{60}{N_p T_s} \text{ rpm} \]  

(2)

The quantization error is a constant, but the relative measurement accuracy depends on the actual angular speed, \( \omega \):

\[ e_{\omega \%} = \frac{2\pi \omega}{\omega N_p T_s} \times 100 \]  

(3)

Equation (3) shows that frequency measurement performs quite well for high speeds, but poorly for low speeds. Relative error is bigger for small encoder resolutions, for small sampling periods and for small angular velocities. Table 1 illustrates the trade-off relationship between our sampling rate and quantization error for \( N_p = 617.4 \).

<table>
<thead>
<tr>
<th>( T_s [\text{ms}] )</th>
<th>( \Delta \omega [\text{rpm}] )</th>
</tr>
</thead>
<tbody>
<tr>
<td>20</td>
<td>4.8591</td>
</tr>
<tr>
<td>10</td>
<td>9.7182</td>
</tr>
<tr>
<td>5</td>
<td>19.4363</td>
</tr>
</tbody>
</table>

In this case, we will choose fastest sampling rate (\( T_s = 5 \text{ ms} \)), because a slower rate would generate more delay to the system, as we will discuss in subsection 2.3. However, we have to accept the bigger quantization error of \( \Delta \omega = 19.4 \text{ rpm} \). The sampling rate could not be made higher due to processing limitations.

2.3 Input Discretization

It is not possible to generate continuous voltage signals with Arduino Nano to excite the motor. Actually, considering the motor as a low-pass filter (as we will show in subsection 2.4), we can use a PWM signal to do this. If the modulation frequency is high enough, we can substitute the PWM signal for its mean value. For our problem, we set this frequency to 4 kHz.

Nonetheless, for digital control purposes, the PWM signal will be updated from time to time. Assuming that this time is the same as the sampling period used to sample the encoder signals, we realize that our input signal is discretized. In fact, this time-discretization effect acts like a zero-order hold (ZOH). It can be shown (Lathi, 2009) that the transfer function of the ZOH is:

\[ ZOH(s) = \frac{1 - e^{-sT_s}}{sT_s} \]  

(4)

In order to understand better the effects of the ZOH over the system, we can look at its Bode diagram, that is displayed in Fig. 3 for \( T_s = 5 \text{ ms} \). The graph shows that the ZOH only affects the gain for high frequencies near the sampling rate (200 Hz, in this case, or 1256 rad/s). On the other hand, the ZOH effect over the phase is considerable even for small frequencies. So, if we use frequencies 5 times smaller than our sampling rate to guarantee Nyquist-Shannon sampling theorem (Lathi, 2009), the ZOH gain effect can be neglected, but the phase must be considered.

2.4 Linear DC Motor Model

A common linear DC motor model used in literature is shown in Fig. 4 (Ogata, 2009). The parameters of the motor are resistance (\( R \)), inductance (\( L \)), viscous friction (\( b \)), electromotive force constant (\( K_b \)), motor torque constant (\( K_t \)) and moment of inertia (\( J \)) of the rotor (without
the wheel or load). The motor torque and electro-motive force are:

\[ T(t) = K_i i(t) \] (5)
\[ e(t) = K_b \dot{\theta}(t) \] (6)

Furthermore, using the law of conservation of energy, we can show that \( K_i = K_b = K \). Using Newton’s laws, we can write the ODE’s:

\[ \dot{J} \dot{\theta} + b \dot{\theta} = K i \] (7)

\[ L \frac{di}{dt} + Ri = V - K \dot{\theta} \] (8)

Finally, applying Laplace Transform, we find the second order speed transfer function of the motor, given by Eq. (9). Note that this is a transfer function of a low-pass filter.

\[ G(s) = \frac{\theta(s)}{V(s)} = \frac{K}{(Js + b)(Ls + R) + K^2} \] (9)

However, all the parameters of the model are unknown, since the manufacturer does not inform them. As a matter of fact, there are two parameters that can be easily measured by a multimeter: \( R \) and \( L \). Their values are displayed in Tab. 2.

Table 2: Values of \( R \) and \( L \) measured by a multimeter.

| \( R \) | 4.5 \( \Omega \) |
| \( L \) | 524 \( \mu \)H |

Assuming we work with frequencies 5 times smaller than the sampling rate (\( \leq 50Hz \)), we can do the following approximation:

\[ |Ls| = 2\pi L f \leq 500 \cdot 10^{-6} \cdot 2\pi \cdot 50 = \]
\[ = 0.1571 \leq R = 4.5 \] (10)
\[ \Rightarrow Ls + R \approx R \]

Therefore, we can reduce the order of our system to first order:

\[ G(s) \approx \frac{K}{R(s + b) + K^2} = \frac{K}{s + \frac{Rb + K^2}{R}} = K' \frac{a}{s + a} \] (11)

where \( K' = \frac{K}{R} \) and \( a = \frac{Rb + K^2}{R} \).

Equation (11) shows that we do not need to know the values of \( K \), \( J \) and \( b \) to determine the transfer function. Instead, finding the system gain, \( K' \), and the pole, \( a \), is enough to model the motor.

### 2.5 Nonlinearities of the System

In subsection 2.4, we presented the linear system as the motor model. Unfortunately, real motors have nonlinearities and, depending on the application, it is necessary to consider them.

An important nonlinearity is the dry friction, which is composed by static friction and kinetic friction (or Coulomb friction). The static friction is present when there is no relative movement between two bodies in contact. Thus, there is a region called dead zone (Fig. 5), where, even if a force is applied, the movement does not start. A breakaway force is needed to put the object in movement. When the motor starts to move, kinetic friction appears and it opposes the direction of movement, as shown in Fig. 5. These non-modeled friction are important, since they generate delay and damping to the system, although they are difficult to be identified. Detailed discussion about friction in motors can be found at (Kara and Eker, 2004) and (Virgala et al., 2013).

Another significant nonlinearity is battery saturation and velocity limitation due to physical construction of the motor. This represents a problem, if, for example, classical control techniques are used to develop a controller or to identify a system using frequency-domain analysis, since these limitations are not considered in these theories. A better theory to deal with these restrictions is the predictive control, as shown by (Klančar and Škrjač, 2007), although we will use only classical control in this paper.

### 3 Experimental Results

#### 3.1 System Identification

To identify the system, we need to find the values of the gain \( K' \) and the pole \( a \) from Eq. (11). We
will work with the Bode diagram, a frequency-domain technique, to find these parameters. A frequency analysis is preferred, instead of time-domain, because the pole position can be identified clearly in the Bode diagram.

To plot the Bode diagram, we used a PWM signal from the microcontroller to simulate the input sine wave. This wave is discretized with the same period of the sampling rate \(T_s = 5\text{ ms}\) and is also quantized, since the PWM of Arduino Nano can assume 255 values of duty cycles. Just before updating the PWM duty cycle, we read a counter variable that saved the number of pulses sent by the encoder since the beginning. This counter is, then, sent through serial communication to a computer that will process the data.

To automatize the data extraction, an embedded program that automatically generates each test frequency was built. For each frequency, we sent many wavelengths in order to reduce the variance of our estimate of the measure. This is important, since our encoder has a considerable quantization error. We used an amplitude of \(V_0 = 5.1\text{ V}\) and frequencies up to 10 Hz.

There are now two possibilities: use position or velocity to plot the Bode diagram. However, we noticed that our motor had a position bias (due to manufacturing issues), which increased linearly over time, as shows Fig. 6. This graph shows the system response to a sine wave of frequency 5 Hz. If we get velocity by differentiating position, the bias becomes a constant and finding the amplitude and phase of the output signal would be easier. Thus, we will plot a velocity Bode diagram.

We will use optimization to determine the amplitude and phase of the output wave. So, defining the error as:

\[
e_s(A, \phi, t) = \omega_{\text{exp}}(t) - A \sin(2\pi ft + \phi)
\]

where, \(\omega_{\text{exp}}(t)\) is the experimental velocity at time \(t = nT_s, n = 0, 1, 2..., N\), \(N\) is the number of samples and \(f\) is the test frequency, we also define the cost function:

\[
J_s(A, \phi) = \sum_{t=0}^{NT_s} c_s^2(A, \phi, t)
\]

We must solve the following optimization problem:

\[
(A^*, \phi^*) = \arg \min \sum_{t=0}^{NT_s} c_s^2(A, \phi, t)
\]

which can be easily solved using \textit{fminsearch} function from Matlab, that uses a simplex search method (Lagarias et al., 1998). An optimization example is shown in Fig. 7.

After repeating the sine optimization for each test frequency, we plot the Bode diagram in Fig. 8. The observed gain plot is characteristic of a first-order system and we can detect a pole near \(40\text{ rad s}^{-1}\), when the amplitude decreases 3 dB. However, near \(40\text{ rad s}^{-1}\) the phase is smaller than \(-45^\circ\), revealing it is not actually of first-order. This phase behavior was expected, if the ZOH effect is considered, since it adds delay to the system (approximately \(-10^\circ\) for 60 rad/s), but it does not modify the gain for small frequencies.

In order to have a better pole estimate, we will also use optimization. However, the value of \(K'\) can easily be obtained by looking at the step \((V_0 = 5.1\text{ V})\) response in steady-state. We found a steady state velocity of 513.2 rpm and a gain of:

\[
K' = \frac{513.2}{V_0} = 100.6
\]
Thus, we only need to determine the pole value. Consider the gain and phase error functions in Eqs. (16) and (17). $A_{\text{opt}}(f)$ and $\phi_{\text{opt}}(f)$ are the points obtained from the Bode diagram and Eq. (18) is our cost function to be optimized:

$$e_g(a, f) = A_{\text{opt}}(f) - 20 \log(|G(s)ZOH(s)|)$$

$$e_\phi(a, f) = \phi_{\text{opt}}(f) - \angle(|G(s)ZOH(s)|)$$

where $G(s)ZOH(s) = \frac{K' a s + 1 - e^{-sT_s}}{sT_s}$

$$J_B(a) = \sum_f e_g^2(a, f) + e_\phi^2(a, f)$$

Hence, we need to solve the following problem:

$$a^* = \arg \min \sum_f e_g^2(a, f) + e_\phi^2(a, f)$$

After solving this optimization problem, we found $a = 36.73$ and:

$$G_{\text{opt}} = \frac{3694.6}{s + 36.73} \frac{1 - e^{-sT_s}}{sT_s} \text{ rpm/V}$$

A comparison between our optimized transfer function and the experimental Bode diagram is shown in Fig. 9. We observe that the experimental gain matches very well with our optimization, but the real motor has more delay than our model. This could be caused by the non-modeled nonlinearities discussed in 2.5.

### 3.2 Model Validation

To validate the model obtained, we propose two alternatives: with open loop and closed loop. If experimental step response is close to the simulated response, then our model is good to be used for the purposes we mentioned.

For the open loop, we compare the experimental step-response with our model ($G_{\text{opt}}$) step-response. Figure 10 illustrates that our model works well for this input. To plot the experimental data and diminish the estimate variance (quantization error), we took the mean over 10 trials.

Then, we propose the speed controller system of Fig. 11 to work as the closed loop system. Differently from before, now we do not have control over the input voltage. In fact, the controller output is not limited, thus we must consider the battery saturation in this model. We also simulated the PWM and the quantization of the encoder. The system is the same as $G_{\text{opt}}$ and the comparison between experimental results and the simulated model is in Fig. 12. The control gain ($k$) value was 0.04 and the reference was 360 rpm.

We observe that simulation and experiment behaviors are alike, even if nonlinear frictions were not considered, although voltage saturation plays an important role at the beginning of the simulation. Therefore, it is necessary to consider this nonlinear effect for control purposes.

The controller also presents a steady-state error for a step input, which is expected for this motor, since it is a system of type zero (the ZOH singularity is removable). Control Theory determines that for an input $R(s)$, the steady-state er-
error for this kind of system is:

\[ e(\infty) = \lim_{s \to 0} \frac{sR(s)}{1 + kG_{opt}(s)} = 71.66 \text{ rpm} \quad (21) \]

The experimental steady-state error was of 72.63 rpm, which is very close to the theoretical.

4 Conclusions

We conclude that with a low cost setup (our own robot), we could identify the system of a hobby DC motor, through the use of a frequency-domain analysis tool (the Bode diagram) and optimization. We found that a first order linear model is good for our motor.

Besides, the model was successfully validated, by comparing the open loop and closed loop performances. One should be careful with nonlinearities, such as the battery saturation, although we did not take into account nonlinear frictions and we still obtained good simulation results.

Future works include developing a model for the entire robot, using the motor model obtained, so as to simulate it or use it for control purposes.

Acknowledgments

We would like to thank ITAndroids and all its members that gave us support, both material and intellectual. We also thank CESAR, Poliedro and Mectron Odebrecht for the financial support.

References


