6.320 Final Project Report
Using Sate Space Modelling for Self-Standing Robot

Dimitris Koutentakis

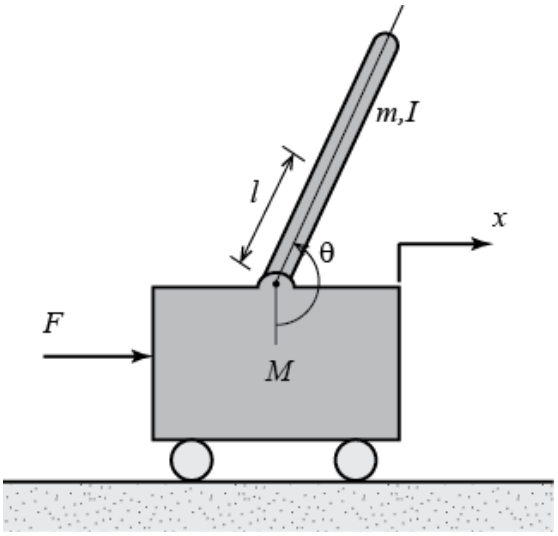
Abstract

In the project, we take a look at the classic "inverted pendulum" controls problem, and implementing a working solution on a `Pololu Balboa` robot platform. We build the robot, design a state-space model and discuss its performance with gains obtained from the LQR method. Our final robot was able to stand for an unlimited amount of time, though stabilization after an external disturbance took time to be achieved.

1 Introduction

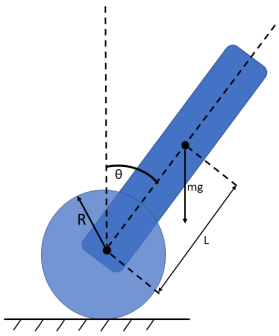
Controlling an upright two-wheeled robot has been a problem of interest for a long time. Be it for commercial applications such as SEGWAY, or military/medical/anthropomorphic robots, the current setup has received a lot of attention and very effective solutions have been found. Here we discuss an implementation of LQR and manual gain tuning in order to effectively control the robot. The problem is part of the more general inverted pendulum problem, described in figure 1.

Figure 1: Inverted Pendulum



According to the body diagram of our problem, which is drawn in figure 2, we develop a state-space model and with the help of `MATLAB` and the LQR method, find appropriate gains for our system.

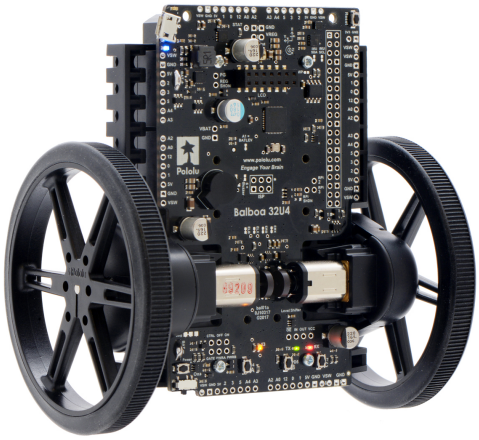
Figure 2: Free Body Diagram



2 Robot

The robot we used was the Pololu Balboa two-wheel robot shown in Fig. 3.

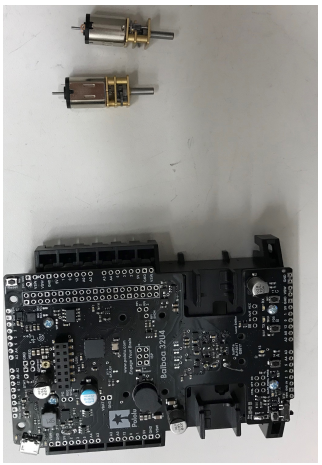
Figure 3: Pololu Balboa Robot



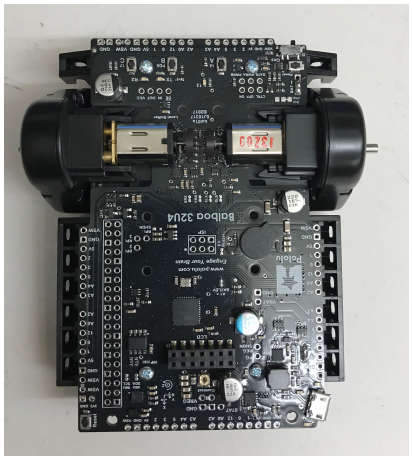
In order to begin working on our project, we first had to assemble the robot which was easily done in the following few steps:

1. Install battery pack - Fig.4a
2. Solder motors - Fig.4b
3. Assemble gears
4. Attach wheels - Fig.4c

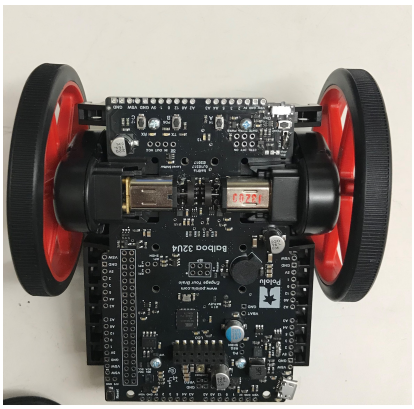
Figure 4: Robot Assmebly Process



(a) Battery Pack Attached



(b) Motors Soldered



(c) Wheels Attached

Furthermore, once assembling the robot, we made all the measurements necessary in order to model our system. The measurements are summarized in table 1.

Symbol	Quantity	Measurement
M_p	Mass of the robot	0.2633
I_p	Moment of inertia of robot	0.000325
l	Half length	0.02
M_w	Wheel mass	0.0198
k_m	Motor torque constant	3.0396
k_e	Motor back EMF constant	0.3183
R	Resistance	20
r	Wheel radius	0.04

Table 1: Motor Measurements

3 State Space Model

After assembling the robot and taking the measurements, we proceeded with the model development in order to find gains for the robot. The states we used are: θ, the angle of the robot as shown above, $\dot{\theta}$, the angular velocity x, the distance and the velocity $\dot{x}$. First, we define the following variables:

$$\beta = 2 * M_w + \frac{2I_w}{r^2} + M_p$$
$$\alpha = I_p\beta + 2M_pl^2(M_w + \frac{I_w}{r^2})$$

With those variables, our state space model will have the following matrices:

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & \frac{2k_mk_e(Mplr - I_p - M_pl^2)}{R\alpha r^2} & \frac{gM_p^2l^2}{\alpha} & 0 \\ 0 & 0 & 0 & 1 \\ 0 & \frac{2k_mk_e(r\beta M_pl)}{(R\alpha r^2)} & \frac{M_pgl\beta}{\alpha} & 0 \end{bmatrix}$$
$$B = \begin{bmatrix} 0 \\ \frac{2k_m(I_p + M_pl^2 - M_plr}{Rr\alpha} \\ 0 \\ \frac{2k_m(M_pl - r\beta)}{Rr\alpha} \end{bmatrix}$$
$$E = \begin{bmatrix} 1 & 0 & -l & 0 \\ 0 & 1 & 0 & -l \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Thus the model can be described with the following equation:

$$\frac{d}{dt}\begin{bmatrix} x \\ \dot{x} \\ \theta \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & \frac{2k_m k_e (Mplr - I_p - M_p l^2)}{R\alpha r^2} & \frac{gM_p^2 l^2}{\alpha} & 0 \\ 0 & 0 & 0 & 1 \\ 0 & \frac{2k_m k_e (r\beta M_p l)}{(R\alpha r^2)} & \frac{M_p g l \beta}{\alpha} & 0 \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \\ \theta \\ \dot{\theta} \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{2k_m (I_p + M_p l^2 - M_p lr}{Rr\alpha} \\ 0 \\ \frac{2k_m (M_p l - r\beta)}{Rr\alpha} \end{bmatrix} V$$

4 First Implementation

For our first implementation, we used the LQR solver from `MATLAB` in the following way:

```
K = lqr(inv(Em)*Am,inv(Em)*Bm,Qp,Rp)
```

The values of Qp and Rp we used are:

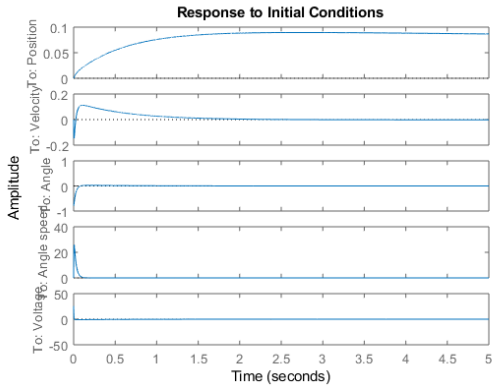
$$Q_p = \begin{bmatrix} 0.025 & 0 & 0 & 0 \\ 0 & 0.01 & 0 & 0 \\ 0 & 0 & 1000 & 0 \\ 0 & 0 & 0 & 0.6 \end{bmatrix}$$

$$R_p = 1$$

The resulting output was:

$$K = \begin{bmatrix} 0.15 \\ 0.1 \\ 32 \\ 0.8 \end{bmatrix}$$

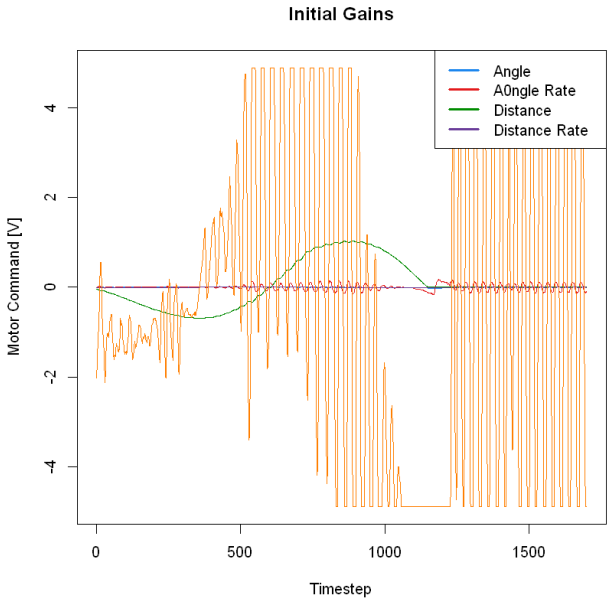
Figure 5: Initial LQR output



Even though the system was not predicted to be entirely stable, (Fig.5) we decided to try it. The robot managed to stay stable for a few seconds, but would quickly start oscillating wildly and get out of control. We first tried implementing a digital filter ($V[t] = 0.9V[t] + 0.1V[t-1]$) in order to reduce the "jitter", but that barely helped.

As seen in Fig. 6, the robot was unable to return to a stable position. Furthermore, what is interesting in the data is how often we reach the maximum voltage of 5V. From this it was obvious that we had to reduce our gains.

Figure 6: Initial Gains Real System Response



5 Final Implementation

In order to make our robot more stable, we decreased the angle gain, and increased the velocity gain, such that our gain matrix was. Performing LQR again with:

$$Q_p = \begin{bmatrix} 0.25 & 0 & 0 & 0 \\ 0 & 170 & 0 & 0 \\ 0 & 0 & 740 & 0 \\ 0 & 0 & 0 & 0.05 \end{bmatrix}$$

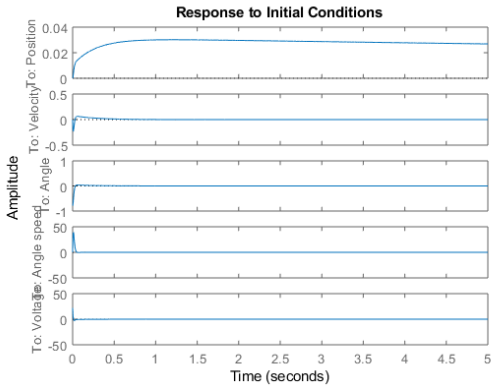
$$R_p = 1$$

And taking into account the rotation of our motors, we get:

$$\begin{bmatrix} 0.5 \\ 7.5 \\ 28 \\ 0.3 \end{bmatrix}$$

The predicted response according to `MATLAB` seemed good 7.

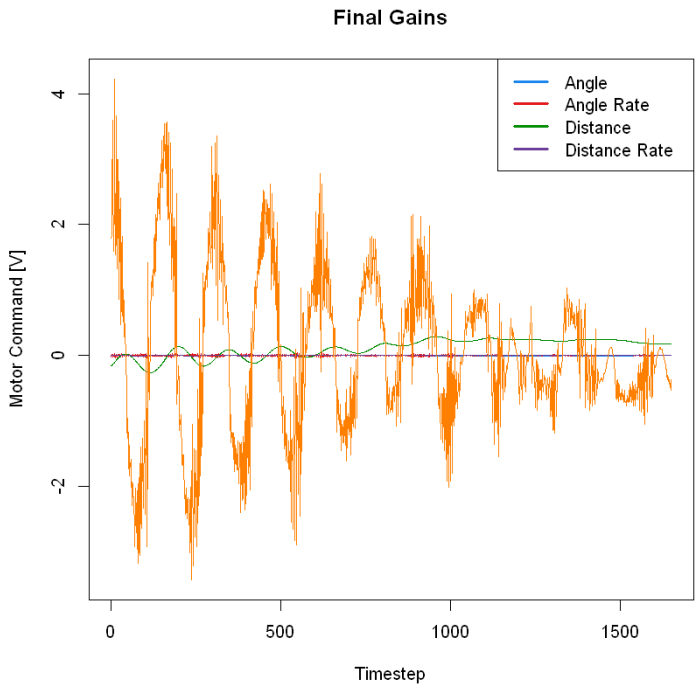
Figure 7: Initial LQR output



Additionally, in order to reduce the jitter, not only did we keep the digital filter, but we also slightly increased our timestep size.

The final robot was able to stand for an indefinite amount of time without much oscillation and was able to recover to stability after external disturbances, as is obvious in figure 8.

Figure 8: Initial Gains Real System Response



6 Conclusion

The inverted pendulum is a very interesting problem in control theory. We attempted to solve it using LQR and we reached a relatively good result. More detailed measurements and accurate motor data, could provide the missing information for an even more stable system. It would also be interesting to see how the system identification method would work for this sort of problem.