Dashpot - Damped Pendulum

Davide Roznowicz*[1,2]

[1]Mathematics Area, mathLab, SISSA
[2]University of Trieste

1 Introduction

In the context of Dashpots, we choose to analyse and stabilise the movement of a damped pendolum. This system is at the basis of several applications in the context of hydraulic components, for example. One main use case is an automatic door that, starting from a completely or partially opened position, is pushed violently by a person or by effect of some air current: our goal is to "gently" accompany this door avoiding it from brutally slamming shut. We will deal with the pendolum thinking of designing this door system.

2 Problem statement

The pendulum is attached to the ceiling. It is made of inextensible wire and connected to a small ball at the extremity. Our setting is such that the pendulum starts from an elevated position: therefore, the wire will be close to the ceiling (horizontal) in the standard settings of experimentation. Then, it will just fall from that starting position and our goal will be to stabilize it some degrees before the impact (the impact point is supposed to be 5 degrees from the set target angle). At that point, when convergence is achieved and angular velocity has decreased by a lot as a consequence, the PID-controlled system can just stop and the door will edge towards the closing position because of a mechanism that gets activated and attracts it to close (this is not related to the project).

We are therefore interested in building a system which, generally speaking, satisfies these requirements (more detail in the appropriate Requirement section below):

- Small overshoot: most important property as we do not want the pendulum (or door) to violently touch any surface before the final mechanism is activated.
- Settling time can be longer than usual modeling problems.
- Oscillations should not be too large as this might cause the pendulum's (or door's) internal automatic system to get damaged.

3 Physics equations

The image 1 gives an overview of the system. The control u, which we intend to apply to stabilize the system, works precisely in the direction of the movement, as in the image: we can imagine a small engine functioning within the ball.

$$m\dot{\omega}r = u - b\omega r - mg\sin(\theta) \tag{1}$$

*droznowi@sissa.it

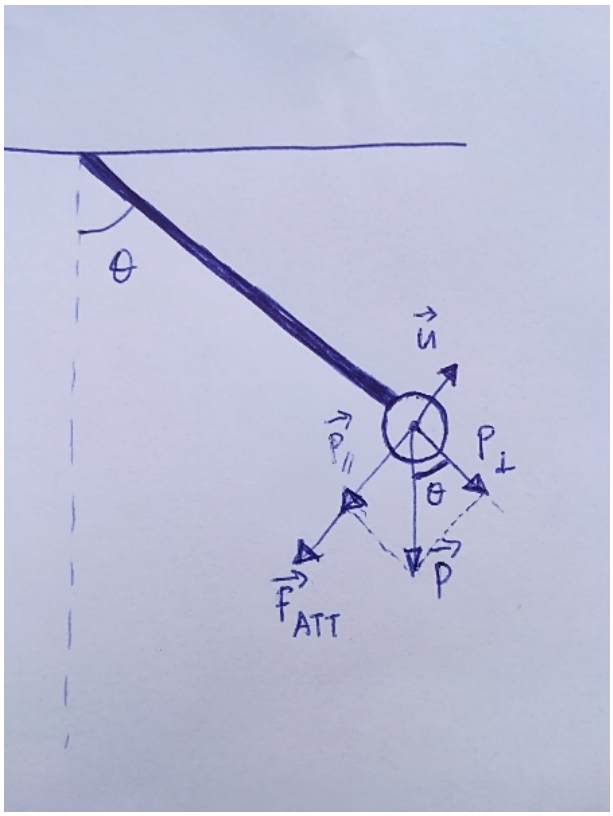

Figure 1: Schema of forces on pendulum

$$\dot{\theta} = \omega \tag{2}$$

$$\dot{\omega} = \frac{u}{mr} - \frac{b\omega}{m} - \frac{g\sin(\theta)}{r} \tag{3}$$

Equation 1 clearly describes the equilibrium of the forces at play. We can see that r describes the radius or length of the wire; b is a friction coefficient; g is the acceleration of gravity; θ is the angle (as shown in the image); m is the mass of the ball; ω is the derivative in time of θ. We can then rewrite it into the usual system of two differential equations.

4 PID control

We perform several runs of the PID control model with several different parameter combination to better assess which setting is suitable for our needs.

4.1 Simulation

The standard setting is described by a starting angle of $\theta = 90$ degrees and $\omega = 0$. The PID constants are manually tuned in a careful way to achieve the desired effects outlined in the *Problem Statement* section:

- We first proceed by increasing K_p until a reasonable error is achieved.
- Then, K_i is increased a little bit to achieve long-term convergence; however, it is not increased much because it would trigger a higher overshoot.
- K_d is the last parameter to be acted upon to regularize the system.

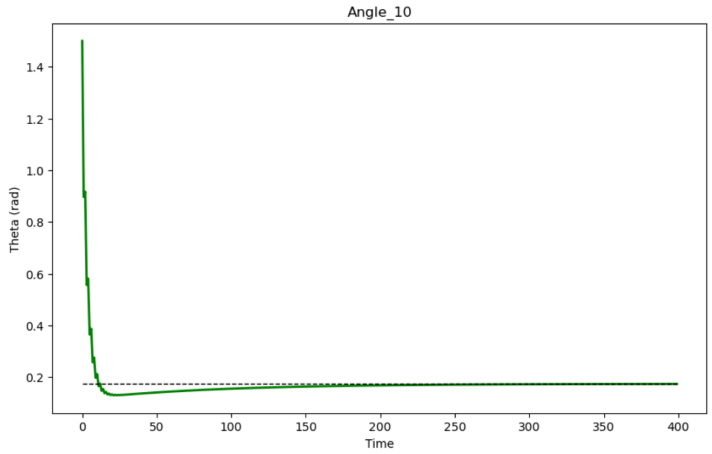

Figure 2: Theta values within the PID-controlled system

	Angle_10
overshoot	0.0400
rise time	11.0000
steady state error	0.0006
settling time	348.0000

Figure 3: Table with main simulation metrics

4.2 Verification of Formal Requirements

These are our requirements, where the second one is the most important for us:

- $\phi_1 = \mathrm{G}[10, 400](abs(\theta(t) - \theta(t-1)) < 0.1)$: would like oscillations not to be too strong because it might provoke some damage to the dashpot mechanism. Our goal is to preserve its correct movement for the long term.
- $\phi_2 = \mathrm{G}[10, 400](abs(\theta(t) - \theta_{target}) < 0.08)$: we want to avoid big overshoots exceeding 5 degrees or 0.08 rads
- $\phi_3 = \mathrm{G}[0, 400](\theta(t) > 0.08)$: we always want the angle to be higher than 5 degrees as we know there is always a hard object at 0 degrees

	Angle_10		Angle_20		Angle_30		Angle_40	
	Min Robustness	Max Robustness	Min Robustness	Max Robustness	Min Robustness	Max Robustness	Min Robustness	Max Robustness
PHI_1	0.053307	0.053307	0.059404	0.059404	0.065502	0.065502	0.071602	0.071602
PHI_2	0.006119	0.006119	0.010198	0.010198	0.014363	0.014363	0.018662	0.018662
PHI_3	0.035587	0.079364	0.039664	0.079425	0.043832	0.079487	0.048139	0.079551

Figure 4: Robustness Table

Since our work is fully done in Python, we use the package Moonlight with STL instructions. The Moonlight functions are used to track the signals. The simple idea is:

- positive values mean that the test has been passed as the signal stays within the boudaries set by the verification requirements
- Negative values mean that the test has failed.

The higher the absolute value of the robustness signals, the stronger the signal (both in case of positive or negative outcome).
Another thing to keep into account when reading the tables is that, when min and max are the same, it happens because Moonlight returns only one value (i.e. "the most exceeding" value).

4.3 Falsification

In this section we stress-test the system to see whether it is robust and reliable enough to handle unexpected situations. We divide the experiments into two groups:

- add perturbations to θ values
- change initial values and increase angular velocity ω

	Angle_10		Angle_20		Angle_30		Angle_40	
	Min Robustness	Max Robustness	Min Robustness	Max Robustness	Min Robustness	Max Robustness	Min Robustness	Max Robustness
PHI_1	0.059496	0.059496	0.065563	0.065563	0.071622	0.071622	0.077675	0.077675
PHI_2	0.019461	0.019461	0.023318	0.023318	0.027275	0.027275	0.031381	0.031381
PHI_3	0.019461	0.066707	0.023318	0.066782	0.027275	0.066866	0.031381	0.066960

Figure 5: Standard setting; relevant noise is added to θ estimations.

	Angle_10		Angle_20		Angle_30		Angle_40	
	Min Robustness	Max Robustness	Min Robustness	Max Robustness	Min Robustness	Max Robustness	Min Robustness	Max Robustness
PHI_1	-0.123359	-0.123359	-0.125276	-0.125276	-0.127179	-0.127179	-0.129065	-0.129065
PHI_2	0.004112	0.004112	0.004632	0.004632	0.005050	0.005050	0.005329	0.005329
PHI_3	0.012601	0.077312	0.015703	0.077383	0.015353	0.077459	0.011214	0.077540

Figure 6: decrease starting angle to $\theta = 60$ degrees and increase ω to 300. Some small noise is added to θ estimations as well.

Some tests have not passed in table 6, all related to ϕ_1. In fact, the system was pushed to the limits on purpose, to have an idea of how it breaks. High angular velocity (ω) and little delta of the angle (difference between start and target) are causing extremely high short-term swings. The second property, regarding the necessity for small overshoot, is still respected.

5 Conclusions

Further PID parameters optimization might help in designing a better system. Several other experiments have been tried in the jupyter notebook, also with different target signals.

References

[1] R. Alur. *Principles of cyber-physical systems.* MIT press, 2015.

[2] C. Baier. Katoen, jp: Principles of model checking. *The MIT Press, London, UK*, 3:1–3, 2008.

[3] A. Platzer. *Logical foundations of cyber-physical systems.* Springer, 2018.