A Working Example of an Adaptive Controller With the Slotine Li Algorithm

Roberto Cadena Vega

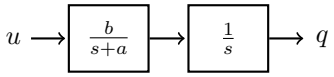

Fig. 2. Simplified servomechanism model

***Abstract*—The abstract goes here.**

I. Introduction

THE intention of this paper is to provide a simple implementation of the Slotine Li Algorithm to control a robotic manipulator with an adaptive control law.

II. Preliminaries

A. Servomechanism Model

The servomechanism used in this experiment consists of a DC motor with a brass disc on it's axis, connected to a power amplifier, so as to have a block diagram representation like the one in the figure 1.

Where β is the gain of the power amplifier because of the armature current feedback loop, K_C is a pre amplification stage, K is the current-torque relation, Kb is related to the counter electromotive force. Let's consider a high gain in the power amplification stage due to the current feedback loop, we can simplify this model to a much more simpler one:

$$\ddot{q}(t) + a\dot{q}(t) = bu(t) \tag{1}$$

Where a and b are defined as:

$$a = \frac{B_m}{J_m} \tag{2}$$

$$b = \frac{K}{K_C J_m} \tag{3}$$

We consider β of great magnitude, so that the time constant of the electric servomechanism is smaller than the mechanical one.

R. Cadena Vega is a student of the Department of Automatic Control, CINVESTAV-IPN, México, e-mail:(rcadena@ctrl.cinvestav.mx).

Manuscript delivered April 20, 2015

B. Parameter Identification

In previous works it has been stablished that the values of the constants a and b are:

$$a = 0.45 \tag{4}$$

$$b = 31.0 \tag{5}$$

So, when presented the results of the estimation of this parameters, they will be compared against this results.

C. Slotine Li Algorithm for Non-linear Robot Manipulators

The Slotine-Li algorithm for robotic manipulator considers a dynamic equation for the robot like:

$$H(q)\ddot{q} + C(q,\dot{q}) + g(q) = \tau \tag{6}$$

Which can be rewritten like this:

$$\tilde{H}(q)\ddot{q}_r + \tilde{C}(q,\dot{q})\dot{q}_r + \tilde{g}(q) = Y(q,\dot{q},\dot{q}_r,\ddot{q}_r)\tilde{a} \tag{7}$$

Where $\hat{a}$ is the parameter estimated through the adaptation law, and $\hat{H}(q)$ is the matrix $H(q)$ substituted with the estimated parameter $\hat{a}$. More over, we define the error in the estimation of the parameter $\tilde{a} = a = \hat{a}$ and Y only depends on dynamic parameters. The error in the estimation of the state of the system is defined as $\tilde{q}(t) = q(t) - q_d(t)$ and we define a new state as:

$$\dot{q}_r = \dot{q}_d - \Lambda\tilde{q} \tag{8}$$

so that when we use the proposed control law:

$$\tau = \hat{H}(q)\ddot{q}_r + \hat{C}(q,\dot{q})\dot{q}_r + \hat{g}(q) - K_D s \tag{9}$$

with the adaptation law:

$$\dot{\hat{a}} = -\Gamma Y^T s \tag{10}$$

Γ being a constant positive definite matrix, and s being defined as:

$$s = \dot{q} - \dot{q}_r = \dot{\tilde{q}} + \Lambda\tilde{q} \tag{11}$$

the equation of the closed loop system will be as stated in 7.

D. Trajectories Generation

The algorithm needs not only the signal for the trajectory to track, but the derivative and the second derivative, both of them can be acheived with a second order filter like the one in the figure3.

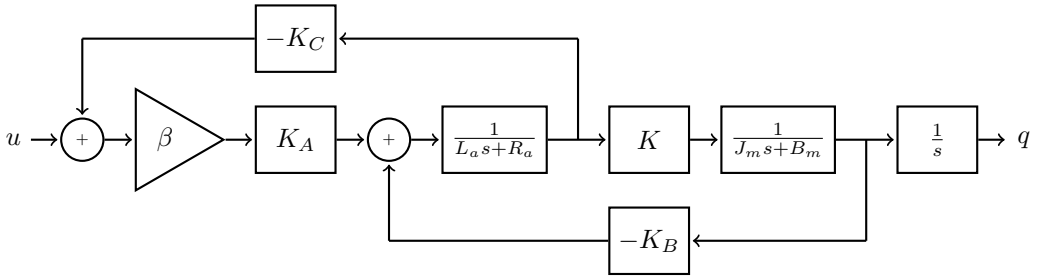

Fig. 1. Servomechanism model

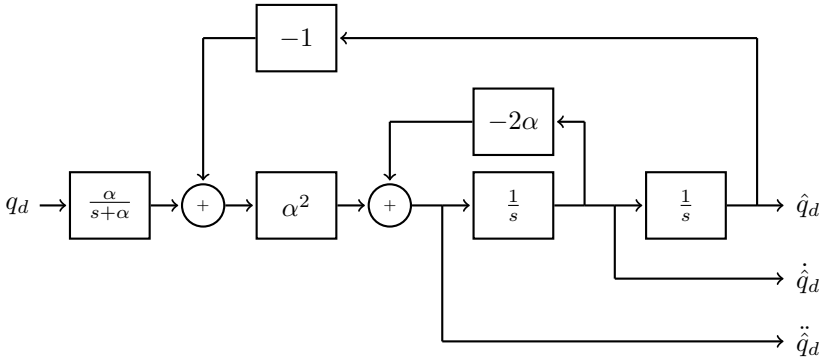

Fig. 3. Second order filter for trajectory generation

III. Control Law Derivation

Since our model is linear and the control law is designed for a non-linear system, it is necessary to derive a simplification for the control law. Rewriting 1 to look like the non-linear manipulator model we have:

$$\frac{1}{b}\ddot{q} + \frac{a}{b}\dot{q} = u \tag{12}$$

And the proposed control law then translates to:

$$u = \frac{1}{\hat{b}}\ddot{q}_r + \frac{\hat{a}}{\hat{b}}\dot{q}_r - K_D s$$

Where we will define $\beta = \frac{1}{\hat{b}}$ and $\alpha = \frac{\hat{a}}{\hat{b}}$ to handle this values easier, so that we will have:

$$u = \beta\ddot{q}_r + \alpha\dot{q}_r - K_D\left(\dot{q} - \dot{q}_r\right) \tag{13}$$

And the adaptation law remains:

$$\dot{\hat{a}} = -\Gamma Y^T s$$

Where the vector of parameters to estimate will be:

$$\hat{a} = \begin{pmatrix}\alpha \\ \beta\end{pmatrix} \tag{14}$$

The vector of known signals is:

$$Y = \begin{pmatrix}\dot{q}_r \\ \ddot{q}_r\end{pmatrix} \tag{15}$$

The control and adaptation laws are implemented in the figures 4 and 5.

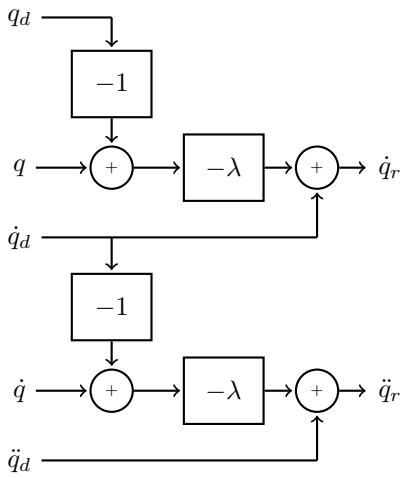

Fig. 4. Slotine-Li algorithm implementation

IV. Experiment Parameters

The control law derived from the Slotine-Li algorithm was implemented in MATLAB-Simulink platform with WINCON from Quanser Consulting as the software link between MATLAB and the Copley Controls PWM power amplifier to feed the control signal to the servomechanism and the BEI optical encoder that is coupled to the servomenchanism shaft to retreive angular position measurements through A Servo To Go Inc. board that actually makes the data acquisition.

The matrix Γ was chosen to be:

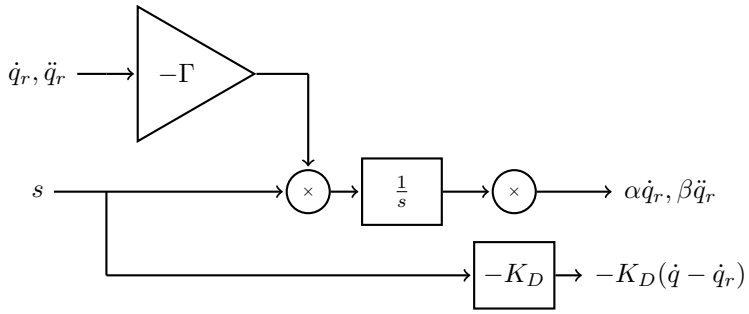

Fig. 5. Adaptation law implementation

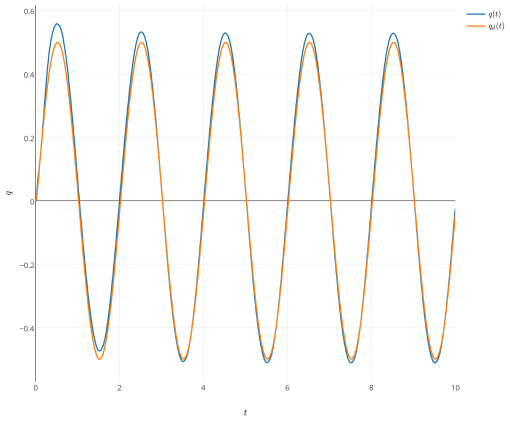

Fig. 6. Servomechanism response against generated signal

$$\Gamma = \begin{pmatrix} 0.5 & 0 \\ 0 & 0.5 \end{pmatrix}$$

and $\lambda = 1$ with a derivative gain of $K_D = 0.5$. The constat for the trajectory generation filter was chosen to be $\alpha = 100$, so that the generated signal is as close as it can be to the desired one.

V. Experimental Results

The experimental results show that while the trajectory tracking was not good, as seen in the figure 6, in the first seconds of the experiment, due to the adaptation law slow convergence of the model parameters, it still manages to reduce the error signal shown in figures 7 and 8. It is noticeable too that the control signal has an initial impulse with a moderate magnitude (figure 10), but afterwards goes on to be bounded to a small magnitude region (figure 9).

VI. Conclusion

While the implementation of this algorithm gives an efficient way of controlling a non-linear o linear system, it demands to be aware of the problems of the gradient descent algorithm which can be unstable for the initial time. It also requires the tuning of Γ and Λ positive definite matrices, which even simplifying to be diagonal matrices, can be difficult to work with if the model is not well known. The benefits of this algorithm is that you can track a signal rather than having a constant signal without knowing the parameters of the system, and this makes the implementation completely worth it.

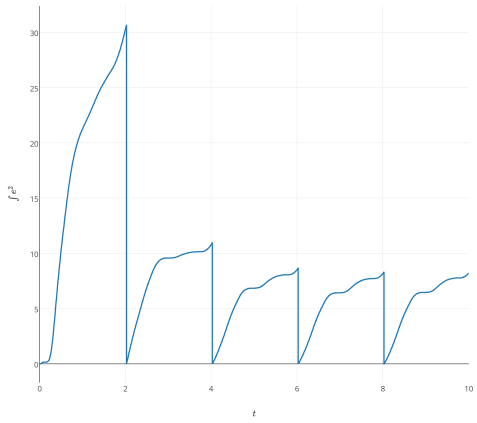

Fig. 8. Integrated square error

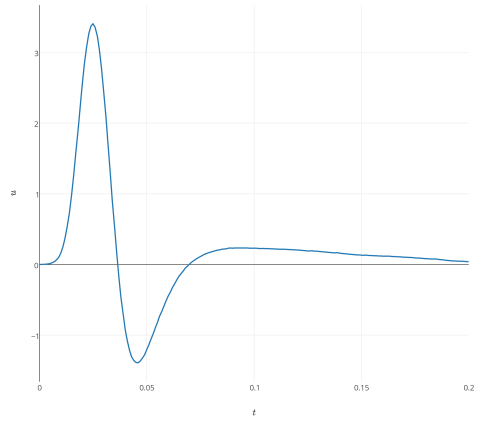

Fig. 10. Control signal first impulse

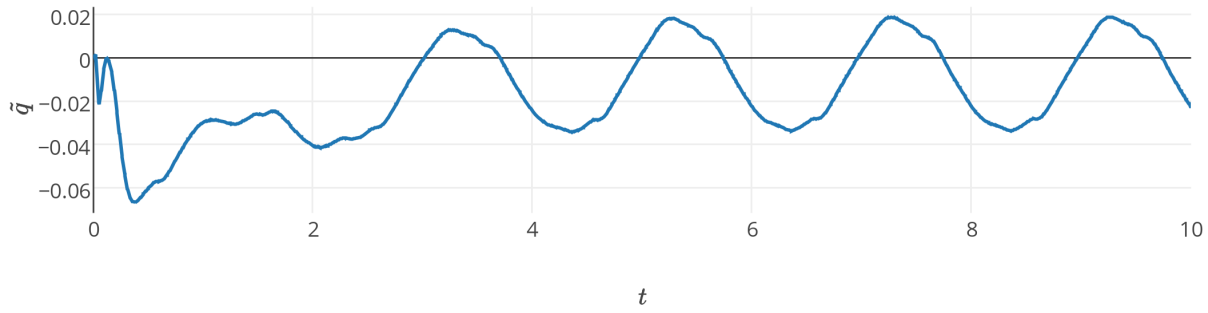

Fig. 7. Error signal

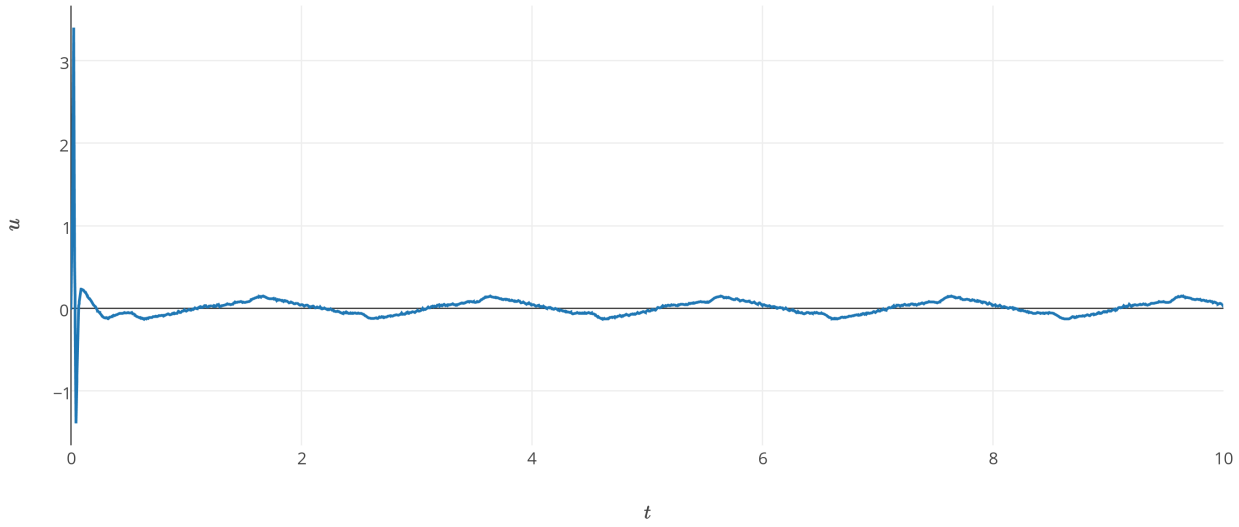

Fig. 9. Control law signal

REFERENCES

[1] H. Kopka and P. W. Daly, *A Guide to LaTeX*, 3rd ed. Harlow, England: Addison-Wesley, 1999.

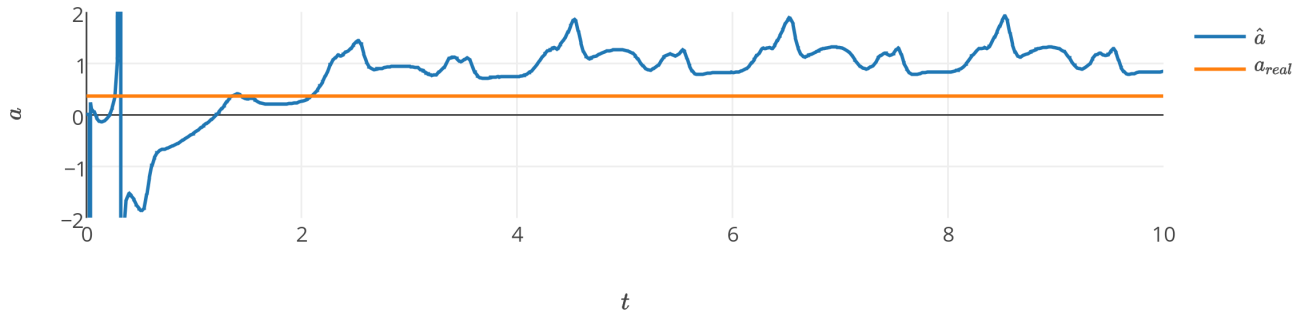

Fig. 11. Estimated signal of parameter a

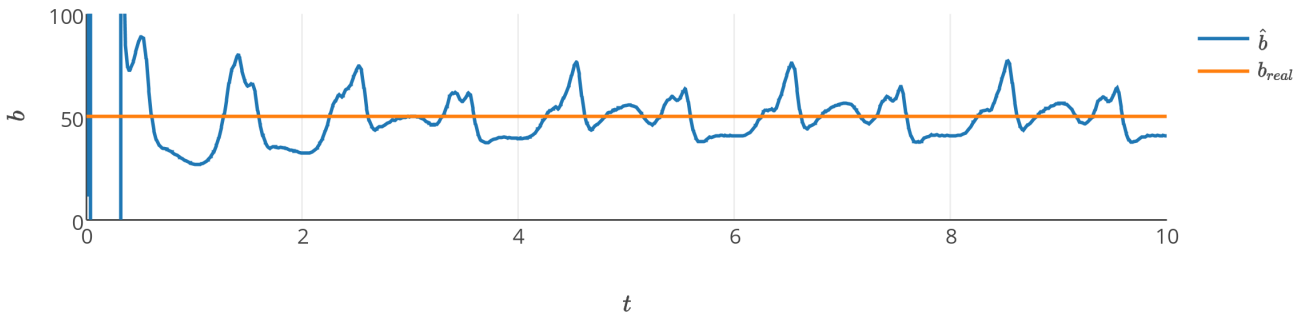

Fig. 12. Estimated signal of parameter b