# Control and Identification Toolbox (CIT): An Android application for teaching automatic control and system identification


## Abstract

This article presents a free Android application, named as Control and Identification Toolbox (CIT), for teaching the automatic control and parameter identification of first and second order linear dynamic systems. The proposed application allows performing real-time experiments with these systems, monitoring their response, and saving data of them using any android device supporting a universal serial bus (USB). Data acquisition is carried out through an Arduino UNO board that allows a fast sampling frequency of up to 500 Hz. PID and model reference adaptive controllers are programed and tuned in the application, whereas the parameter identification is performed by means of the Recursive Least Squares Method. Experimental results obtained using first and second order low pass filters confirm the effectiveness of the proposed application.



## Keywords

Experimental design, educational tool, generating examples


## 1 Introduction

Discrete PID controller [1]. Testing citation [2]. The app is available at [3]

# 2 Bridge device

A bridge device is required to 1) translate the output generated by the algorithms into real world voltage levels and 2) capture the voltage levels of the output of the plant to feed to the android device. Authors used an Arduino Uno board as a bridge device between the Android system and the plant $G(s)$. In this section the bridge circuit's hardware and firmware requirements are presented.

## 2.1 Hardware

The bridge device requires one analog input, one analog output and an USB communication implementing RS232 protocol. The universal serial bus (USB) communication is selected over the Bluetooth because the physical connection between can also be used to power the bridge device and the plant. The bridge circuit along with a plant is shown in Figure (Fig:BridgeCircuit). The operational amplifier KA358 is optional, which serves as an amplifier with unit gain for the analog output of the bridge device. Characteristics of the bridge hardware is given in Table 1.

## 2.2 Firmware

The algorithm shown in Figure (Fig:Firmware) is developed with the help of Arduino IDE. The bridge device's USB is configured as a device with communications device class (CDC), which lets the USB host to identify the bridge device as a Recommended Standard 232 (RS-232) port.

The bridge device stays idle until it receives a message. If the first part of the message is 1, it sets bridge device's analog output to the second part of the message. If the first part of the message is 2, it reads analog voltage and sends the read value to the USB host.

# 3 Android application

The user interface is achieved with the help of the android application (app), Control Toolbox (Time domain) [4], which is developed using Java language in the Android Studio environment. Table 3 gives the list of classes used in this app.

## 3.1 Graphical user interface

The graphical user interface (GUI) is kept minimalistic with native android studio elements except for the graphs and seekbars, which are obtained with the help of open-source software GraphView [5] and IndicatorSeekBar(https://github.com/warkiz/IndicatorSeekBar). Figure (Fig:HomeScreen) shows the home screen of the application, which contain several buttons. Pressing a button on the home-screen changes the from home screen to the view associated to it. The following views are available in this GUI.

### 3.1.1 Documentation view

The Documentation view is displayed when the DOCUMENTATION button is pressed on the home screen. Bridge diagram connections to first order and second order systems are shown. displays connection diagrams for the bridge circuit.

### 3.1.2 Settings view

The settings view can be invoked from any view by tapping on the button with gear icon. This button is disabled while controller is activated. The settings view allows the user to modify the following settings: 1. Sampling time, 2. No of samples to retain for plotting the graphs, 3. X-axis window size, 4. Height of the graph. It uses an SQLite database to store the settings information.

### 3.1.3 Controller view

The controller view is shown when a user taps on one of the available buttons for the controllers. The function **GenerateViewFromModel()** generates the controller view associated to the selected **Model**. Pressing a button on the main screen prepares a model associated to the button. The following models are made available in this version of the application: (1) Open loop, (2) Proportional–integral–derivative (PID) controller, (3) First order adaptive controller, (4) First order system identification, (5) Second order system identification, and (6) System identification of first order with integral controller which are associated with the functions (1) **PrepareOpenLoopModel()**, (2) **PreparePIDModel()**, (3) **PrepareFirstOrderAdaptiveControlModel()**, (4) **PrepareFirstOrderIdentification()**, (5) **PrepareSecondOrderIdentification()**, and (6) **PrepareFirstOrderWithControllerIdentification()** respectively. If the bridge circuit is attached to the device with proper permissions, the run button will be enabled.

Figure (Fig:PIDView) shows android app's screen-shot for a closed loop control system with PID controller, which be decomposed into the following six sections.

1. Diagram: Displays the control block diagram and describes the control algorithm.
2. Sampling time: Lets the user to change the sampling time directly from this screen without any need to go to the settings window. This lets the student to change the sampling time in the real-time while the algorithm is running.
3. Parameters: Various parameters of the system can be modified here. These changes reflect in the real-time.
4. Signal generator: Signal generators are used to introduce the analog signal into the system. A set of three signals are allowed for each analog input, which lets the user to generate a custom

signal. For example: the PID controller shown in Figure (Fig:PIDView) analog input for the reference as $r(t) = r_1(t) + r_2(t) + r_3(t)$, where $r_i(t)$, $i = 1, 2, 3$. The app lets the users to select any of the following wave functions for the $r_i$.

- Step

$$r_i = H\left(t - t_{0,i}\right)\left[E_i + \kappa\right] \tag{1}$$

$$H(t) = \begin{cases} 0 & \text{if} \quad t < 0 \\ 1 & \text{if} \quad t \geq 0 \end{cases} \tag{2}$$

where $E_i$ is the signal amplitude, $t_{0,i}$ is time at which the signal starts and $\kappa$ is the offset.

- Sine

$$r_i(t) = H\left(t - t_{0,i}\right)\left[E_i \sin\left(\omega\left(t - t_{0,i}\right)\right) + \kappa\right] \tag{3}$$

where $\omega_i = 2\pi f_i$ is the angular velocity and $f_i$ is the frequency.

- Sawtooth

$$r_i = H\left(t - t_0\right)\left[2E_i f_i \bmod\left(t - t_0, \frac{1}{f_i}\right) + \kappa\right] \tag{4}$$

where $\bmod\left[m, n\right]$ gives the remainder of the division $\frac{m}{n}$ and $f_i$ is the signal frequency.

- Triangular

$$r_i = H_i\left(t - t_{0,i}\right)\left[\frac{2E_i}{\pi}\sin^{-1}\left[\sin\left(\omega_i\left(t - t_{0,i}\right)\right)\right] + \kappa\right] \tag{5}$$

- Square

$$r_i = H_i\left(t - t_{0,i}\right)\left[\operatorname{sgn}\left[\sin\left(\omega_i\left(t - t_{0,i}\right)\right)\right] + \kappa\right] \tag{6}$$

$$\operatorname{sgn}(t) = \begin{cases} -1 & \text{if } t < 0 \\ 1 & \text{if t} \geq 0 \end{cases} \tag{7}$$

- Rectangular

$$r_i = H\left(t - t_0\right)\left[\operatorname{sgn}\left[\sin\left(2\pi f_0\left[\left(t - t_0\right) + \frac{1}{2f_0}\left(0.5 - D\right)\right]\right) - \sin\left(\pi\left(0.5 - D\right)\right)\right] + \kappa\right] \quad (8)$$

where $D$ is the signal's duty cycle.

5. Figure: In this section, the plots to the selected signals and the identified parameters in the real-time. There may be more than one figure for a screen. The height of the figures can be adjusted from the settings. These figures let the student to identify the trend to estimate the properties such as stable, unstable, conversing and diverging etc.

6. Instantaneous values: This section displays the instantaneous values of the selected signals and the identified parameters, which will help the student to write it down in the notes instead of estimating them from the figures.

## 3.2 Communication with the bridge circuit

Android app uses the Arduino Library \url{https://github.com/OmarAflak/Arduino-Library} which is based on the UsbSerial \url{https://github.com/felHR85/UsbSerial} to communicate with the bridge circuit. This library allowed to implement the event-listeners for the port activities which lets to identify the bridge circuit attach, detach, user permission to let the app to communicate with the serial device.

The app can detect the presence of bridge circuit and the app enables the run button only when the bridge circuit is attached and has proper permissions to use it.

### 3.3 Implementing real-time algorithms

In this android app, the real-time execution is performed in a separate background thread with the help of the **doInBackground(Params...)** method of the *AsyncTask*; and the UI's figures and instantaneous values are updated using the **onProgressUpdate(Progress...)** method [6]. Algorithm for

Figure (Fig:RealTimeAlgorithm) shows the implementation of the control algorithms in the real-time.

# 4 Control systems used by the CIT

The proposed Android application allows obtaining the system response of first and second order linear systems controlled in open loop or in closed-loop. Let $u(t)$ and $y(t)$ the input and output of these systems, respectively. An open and a closed-loop systems are shown in Figures (Fig:BlockDiagramOpenLoop) and (Fig:BlockDiagramClosedLoop), respectively. In these Figures, signal $r(t)$ is the reference input to the control systems. Note that in an open loop system $r(t) = u(t)$. Moreover, in the closed-loop system, signal $e(t)$ is the difference between $r(t)$ and $y(t)$. This difference enters to a controller, which produces a control signal that reduces $e(t)$ to zero or to a small value. Through the CIT Android application the following controllers can be programed:

- Proportional Integral Derivative controller (PID).
- Model reference adaptive controller (MRAC).

These controllers are discussed in detail in Sections xx and xx, respectively.

# 5 Mathematical model of first and second order systems

The Android application allows teaching the automatic control and parameter identification of first and second order linear dynamic systems, whose mathematical model is described through the equations (9) and (10), respectively.

$$\dot{y}(t) + \alpha_0 y(t) = \alpha_1 u(t) \tag{9}$$

$$\ddot{y}(t) + \beta_1 \dot{y}(t) + \beta_0 y(t) = \beta_2 u(t) \tag{10}$$

where $\alpha_0$, $\alpha_1$, $\beta_0$, $\beta_1$, and $\beta_2$ are positive parameters. The transfer functions $H(s)$ of systems (9) and (10) are given by:

$$\frac{Y(s)}{U(s)} = H(s) = \frac{\alpha_1}{s + \alpha_0} \tag{11}$$

$$\frac{Y(s)}{U(s)} = H(s) = \frac{\beta_2}{s^2 + \beta_1 s + \beta_0} \tag{12}$$

where $Y(s) = \mathcal{L}[y(t)]$ and $U(s) = \mathcal{L}[u(t)]$, and $\mathcal{L}[\cdot]$ is the Laplace operator.

The performance of the application is experimentally verified using two simple circuits, composed by a first and a second order low pass filters, which are shown in Figures (Fig:CircuitRC1) and (Fig:CircuitRC2). The parameters $R$ and $C$ in these Figures denote resistance and capacitance, respectively.

The mathematical model of the first order low pass filter in Figure (Fig:CircuitRC1) is given by:

$$\frac{Y(s)}{U(s)} = \frac{\alpha_1}{s + \alpha_0} = \frac{\frac{1}{T}}{s + \frac{1}{T}} \tag{13}$$

where $\alpha_0 = \alpha_1 = 1/T$, and $T = RC$ is called time constant; moreover, $u(t)$ and $y(t)$ are the input and the output voltage of the filter, respectively.

On the other hand, the second order low pass filter, shown in Figure (Fig:CircuitRC2), is described through the following mathematical model:

$$\frac{Y(s)}{U(s)} = \frac{\beta_2}{s^2 + \beta_1 s + \beta_0} = \frac{\frac{1}{T^2}}{s^2 + \frac{3}{T}s + \frac{1}{T^2}} \tag{14}$$

where $\beta_0 = \beta_2 = 1/T^2$, and $\beta_1 = 3/T$.

All the experiments, shown in this paper, use the parameters $R = 100$ k$\Omega$ and $C = 1$ $\mu$F, thus obtaining $T = 0.1$ s, $\alpha_0 = \alpha 1 = 10$ s$^{-1}$, $\beta_0 = \beta_2 = 100$ s$^{-2}$, and $\beta_3 = 30$ s$^{-1}$. Moreover, the sampling time $T_s$ defined in the CIT app for all experiments is $T_s = 0.02$ s.

# 6 Responses plotted by the Application

In this section the responses of linear and second order systems, produced by step and sine wave inputs, are plotted by the Android application. This inputs are selected because they are commonly used as test input signals. The reason is that the step input allows to determine how fast the system response can reach the amplitude of the input. On the other hand, the sine wave input allows determining the frequency response of linear systems.

## 6.1 First order system

Applying the step function $u(t) = EH(t)$ to the linear filter in (13) produces [7]:

$$y(t) = E[1 - e^{-t/T}] \tag{15}$$

Fig. x shows the response $y(t)$ plotted by the Android application using $E = 2.5$. Note that at $t = T$ the response $y(t)$ reaches $63.2\%$ of the amplitude $E$ of the input.

On the other hand, a sine wave input with offset $r(t) = F_1 \sin(\omega t) + \kappa$ analytically produces the steady-state response $y_{ss}$ in (16), that is obtained after a large period of time of $y(t)$.

$$y_{ss}(t) = \frac{F}{\sqrt{1 + T^2\omega^2}} \sin(\omega t - \tan^{-1} T\omega) + \kappa \tag{16}$$

Note that the frequency $\omega = 1/T$ is called cut-off frequency and will be denoted as $\omega_c$. Figures x (a) and x (b) present the responses $y_{ss}(t)$ obtained with the ICT Application using $r(t) = 2.5\sin(\omega t) + x$, where $\omega = \omega_c = 0.1$ rad/s and $\omega = 100\omega_c = 10$ rad/s, respectively. It is shown that the the amplitude of the steady-state response $y_{ss}(t)$ in Fig. X (b) is smaller that the one presented in Fig. X (a); moreover, the phase delay of the response $y_{ss}(t)$ in Fig. X (b) is larger than the one of the response $y_{ss}(t)$ in Fig. X (a). Note that using the responses $y_{ss}(t)$ plotted with the app and obtained with the input $r(t) = 2.5\sin(\omega t) + x$, where $\omega$ is varied from $\omega = 0.1$ to $\omega = 10$ with increments of 0.5 rad/s, produces the frequency response of the system, which is shown in Fig. xx

## 6.2 Second order system

The transfer function of the second order filter (14) can be rewritten in the following standard form of a second-order system

$$\frac{Y(s)}{U(s)} = \frac{\omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2} \tag{17}$$

where $\omega_n = 1/T$, and $\zeta = 3/(2\omega_n T)$. Parameters $\omega_n$ and $\zeta$ are named as undamped natural frequency and damping ratio, respectively.

The step response of system (17) depends on the parameters $\zeta$ and $\omega_n$. If $0 < \zeta < 1$, the system is called underdamped and has an oscillatory transient response. When $\zeta = 1$ and $\zeta > 1$ the system response does not have oscillations and is called critically damped and overdamped, respectively. The second order filter (14) is an underdamped system, whose response produced by an input step $u(t)=EH(t)$ is mathematically given by [7]

$$y(t) = E\left(1 + \frac{\omega_n}{2\sqrt{\zeta^2 - 1}}\left[\frac{e^{-\delta_1 t}}{\delta_1} - \frac{e^{-\delta_2 t}}{\delta_2}\right]\right) \tag{18}$$
$$\delta_1 = \zeta + \sqrt{\zeta^2 - 1}, \qquad \delta_2 = \zeta - \sqrt{\zeta^2 - 1}$$

The step response $y(t)$, plotted by the CIT app with $E = 2.5$, is shown in Fig. x. Note that $y(t)$ converges to $E$ in xx seconds.

By applying the sine wave input with offset $r(t) = F\sin(\omega t) + \kappa$ to the second order filter yields the next steady-state response $y_{ss}(t)$:

$$y_{ss}(t) = \frac{F}{\sqrt{\left(1 - \frac{\omega^2}{\omega_n^2}\right)^2 + \left(\frac{2\zeta\omega}{\omega_n^2}\right)^2}} \sin\left(\omega t - \tan^{-1}\left[\frac{\frac{2\zeta\omega}{\omega_n}}{1 - \frac{\omega^2}{\omega_n^2}}\right]\right) + \kappa \tag{19}$$

The response $y_{ss}(t)$, obtained with the ICT Application using $r(t) = 2.5\sin(\omega t) + \kappa$, $\omega = \pi$ rad/s, and $\kappa = x$, is depicted in Fig. X (a). Moreover, Fig. X (b) shows the frequency response of the system corresponding to $r(t) = 2.5\sin(\omega t) + x$, $\omega \in [0.1 - 10]$ rad/s.

# 7 Parameter identification

The CIT Android application permits estimating the parameters of the continuous-time models in equations (9) and (10), as well as the parameters of discrete-time models corresponding to (9) and (10). First, the parameters of the discrete time models are estimated using the Recursive Lest Squares Method (RLSM). Afterwards, these models are converted to their continuous-time counterpart in order to compute the parameters of the continuous-time models.

## 7.1 First order system

The zero-order-hold discretization method allows obtaining the following discrete time model corresponding to (9):

$$y(k) = e^{-\alpha_0 T_s} y(k-1) + \left( \int_0^{T_s} e^{-\alpha_0 \tau} d\tau \right) \alpha_1 u(k-1) \tag{20}$$

where parameters $\alpha_0$ and $\alpha_1$ are suppose to be unknown and will be estimated.

Equation (20) can be rewritten as:

$$y(k) = \theta_1 y(k-1) + \theta_2 u(k-1) = \phi^T(k)\boldsymbol{\theta} \tag{21}$$

$$\begin{aligned} &\theta_1 = e^{-\alpha_0 T_s}, \qquad \theta_2 = \int_0^{T_s} e^{-\alpha_0 \tau} = \frac{1}{\alpha_0}\left[1 - e^{-a_0 T_s}\right] = \frac{1}{\alpha_0}\left[1 - \theta_1\right] \\ &\boldsymbol{\theta} = [\theta_1, \theta_2]^T, \qquad \phi(k) = [y(k-1), u(k-1)]^T \end{aligned} \tag{22}$$

Expression (21) is called linear parameterization, since it is simply a linear equation in terms of the unknown vector $\boldsymbol{\theta}$ [8].

The CIT Application uses the RLSM in order to estimate the vector parameter $\boldsymbol{\theta}$ in (21). The RLSM is given by:

$$\begin{aligned}\boldsymbol{\theta}(k) &= \boldsymbol{\theta}(k-1) + \mathbf{P}(k)\boldsymbol{\phi}(k)\epsilon(k)\\ \mathbf{P}(k) &= \frac{1}{\gamma}\left[\mathbf{P}(k-1) - \frac{\mathbf{P}(k-1)\boldsymbol{\phi}(k)\boldsymbol{\phi}(k)^T\mathbf{P}(k-1)}{\gamma + \boldsymbol{\phi}^T(k)\mathbf{P}(k-1)\boldsymbol{\phi}(k)}\right]\\ \epsilon(k) &= y(k) - \boldsymbol{\phi}^T(k)\boldsymbol{\theta}(k-1)\end{aligned} \tag{23}$$

where $\boldsymbol{\theta}(k) = [\hat{\theta}_1(k), \hat{\theta}_2(k)]^T$ is an estimate of $\boldsymbol{\theta}$, $\gamma$ is called forgetting factor and satisfies $0 < \gamma \leq 1$. Moreover, variable $\mathbf{P}(k) = \mathbf{P}^T(k)$ is called covariance matrix.

**Remark:** The estimate vector $\boldsymbol{\theta}(k)$ converges to $\boldsymbol{\theta}$ if and only if the input $u(t)$ has at least $n/2$ different frequencies, where $n$ is the number of components vector $\boldsymbol{\theta}$ [9].

According to the last remark, the signal $u(t)$ must have at least one frequency so that $\boldsymbol{\theta}$ converges to $\boldsymbol{\theta}$.

From equation (22) it is possible to compute the parameters $\alpha_0$ and $\alpha_1$ of the continuous time model (9) as follows:

$$\hat{\alpha}_0(k) = -\frac{\ln(\hat{\theta}_{1^*}(k))}{T_s}, \qquad \hat{\alpha}_1(k) = \frac{\hat{\alpha}_0(k)\theta_2(k)}{1-\theta_1(k)} \tag{24}$$

$$\hat{\theta}_{1^*}(k) = \begin{cases}\hat{\theta}_1(k) & \text{if} \quad \hat{\theta}_1(k) > 0.01\\ 0.01 & \text{if} \quad \hat{\theta}_{1^*}(k) \leq 0.01\end{cases}$$

where $\hat{\theta}_{1^*}(k)$ is a parameter projection of $\hat{\theta}_1(k)$, that takes only positive values.

An experiment is carried out to identify the parameters $\alpha_0 = 10$ and $\alpha_1 = 10$ of the linear filter in (13). Figures xx and xx show the estimates $\hat{\alpha}_0(t)$, $\hat{\alpha}_1(t)$ and $\hat{\theta}_1(t)$, $\hat{\theta}_2(t)$ provided by the CIT Application,

respectively. It is observed that the estimates $\hat{\alpha}_0(t)$, $\hat{\alpha}_1(t)$ and $\hat{\theta}_1(t)$, $\hat{\theta}_2(t)$ converge close to the nominal values $\alpha_0(t)$, $\hat{\alpha}_1(t)$ and $\theta_1(t)$, $\theta_2(t)$, respectively.

## 7.2 Second order system

Applying the zero-order-hold discretization method to the continuous-time model in (10) produces:

$$\begin{aligned} y(k) &= \theta_1 y(k-1) + \theta_2 y(k-2) + \theta_3 u(k-1) + \theta_4 u(k-2) = \phi^T(k)\boldsymbol{\theta} \\ \boldsymbol{\theta} &= [\theta_1, \theta_2, \theta_3, \theta_4]^T, \qquad \phi(k) = [y(k-1),\ y(k-2),\ u(k-1),\ u(k-2)]^T \end{aligned} \tag{25}$$

An on-line estimate $\boldsymbol{\theta}(k) = [\hat{\theta}_1(k), \hat{\theta}_2(k), \hat{\theta}_3(k), \hat{\theta}_4(k)]^T$ of the parameter vector $\boldsymbol{\theta}$ in (25) is determined through the RLSM given in (23). Since four parameters are estimated, convergence of $\boldsymbol{\theta}$ to $\boldsymbol{\theta}$ requires that $u(t)$ has at least two different frequencies.

For estimating the parameters $\beta_0$, $\beta_1$ and $\beta_2$ of the continuous-time model (10), the following procedure is carried out:

- In each sample time the following matrices are computed

$$\mathbf{A}_d = \begin{bmatrix} 0 & 1 \\ -\hat{\theta}_2 & -\hat{\theta}_1 \end{bmatrix}, \qquad \mathbf{B}_d = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \qquad \mathbf{C}_d = [\hat{\theta}_4, \hat{\theta}_3] \tag{26}$$

  which correspond to the state space representation of the estimated model of (25) given as

$$\begin{aligned} \mathbf{x}(k+1) &= \mathbf{A}_d \mathbf{x}(k) + \mathbf{B}_d u(k) \\ y(k+1) &= \mathbf{C}_d \mathbf{x}(k) \end{aligned} \tag{27}$$

- Matrices in (26) are used to obtain

$$\mathbf{A}_c = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} = \frac{2}{T}\mathbf{R}\left[\mathbf{I}_{2x2} - \frac{8}{21}\mathbf{R}^2 - \frac{4}{105}\mathbf{R}^4\right]\left[\mathbf{I}_{2x2} - \frac{5}{7}\mathbf{R}^2\right]^{-1}$$

$$\mathbf{B}_c = \begin{bmatrix} b_{11} \\ b_{21} \end{bmatrix} = \mathbf{A}_c\left[\mathbf{A}_d - \mathbf{I}_{2x2}\right]^{-1}\mathbf{B}_d, \qquad \mathbf{C}_c = [c_{11}, c_{12}] = \mathbf{C}_d \tag{28}$$

$$\mathbf{R} = \left[\mathbf{A}_d - \mathbf{I}_{2x2}\right]\left[\mathbf{A}_d + \mathbf{I}_{2x2}\right]^{-1}$$

Matrices $\mathbf{A}_c$, $\mathbf{B}_c$, and $\mathbf{C}_c$ correspond to the following continuous-time counterpart of (27)

$$\dot{\mathbf{x}}(t) = \mathbf{A}_c x(t) + \mathbf{B}_c u(t) \tag{29}$$

$$y(t) = \mathbf{C}_c \mathbf{x}(t) \tag{30}$$

where $\mathbf{A}_c$, $\mathbf{B}_c$, and $\mathbf{C}_c$ are computed using the Geometric series method [10].

- Finally, in each sample time the estimates $\hat{\beta}_0$, $\hat{\beta}_1$ and $\hat{\beta}_2$ are computed as follows

$$\hat{\beta}_0(k) = a_{11}a_{22} - a_{12}a_{21}, \quad \hat{\beta}_1 = -(a_{11} + a_{22}), \quad \hat{\beta}_2 = c_{11}(b_{21}a_{12} - b_{11}a_{22}) + c_{12}(b_{11}a_{21} - b_{21}a_{11}) \tag{31}$$

where equations in (31) result by equaling the transfer function $H(s)$ in (12) with the transfer function

$$H(s) = \mathbf{C}_c(s\mathbf{I} - \mathbf{A}_c)^{-1}\mathbf{B}_c \tag{32}$$

The CIT app is used to estimate the parameters $\theta_1$, $\theta_2$, $\theta_3$, $\theta_4$, $\beta_0$, $\beta_1$, and $\beta_2$, of the second-order filter. Figures xx shows the parameter estimates $\hat{\theta}_1$, $\hat{\theta}_2$, $\hat{\theta}_3$, and $\hat{\theta}_4$ of the discrete-time model corresponding to the filter. Furthermore, Fig. X depicts the parameters $\hat{\beta}_0$, $\hat{\beta}_1$ and $\hat{\beta}_2$ of the continuous-time model. Note that all the parameter estimates converge close to their nominal values.

# 8 Automatic controllers

This section describes the Proportional Integral Derivative (PID) controller and the Model Reference Adaptive Controller (MRAC), that are available in the CIT application for controlling fist and second order linear systems. The difference between them and their implementation in the CIT app is also discussed.

## 8.1 Proportional Integral Derivative (PID) controller

Figure (Fig:BlockDiagramPID) shows a closed-loop system, where the system is corrupted by a disturbance $d(t)$ and is controlled through a PID controller defined as:

$$u(t) = K_P e(t) + K_I \int_0^t e(\tau) d\tau + K_D \dot{e}(t), \qquad e(t) = r(t) - y(t) \tag{33}$$

The disturbance $d(t)$ is given by:

$$d(t) = d_1 + d_2 \xi(t) \tag{34}$$

where $d_1 \in R$ is a constant disturbance, $\xi(t)$ is a zero mean white noise, and $d_2$ is its power.

The PID controller is implemented in the CIT app in the following discrete-time versión

$$\begin{gathered} u(k) = u(k-1) + \lambda_1 e(k) + \lambda_2 e(k-1) + \lambda_3 e(k-2) \\ \alpha_1 = K_P + \frac{K_I T_s}{2} + \frac{K_D}{T_s}, \qquad \alpha_2 = -K_P + \frac{K_I T_s}{2} - \frac{2K_D}{T_s}, \qquad \alpha_3 = \frac{K_D}{T_s} \end{gathered} \tag{35}$$

where the integral and derivative terms are approximated by means of the Trapezoidal and the Backward Euler methods [11], respectively.

In the next subsections is analyzed the stability of the closed-loop system (\ref{fig:pid}).

### 8.1.1 First order system

If the plant of the closed-loop system in Figure (Fig:BlockDiagramPID) is the first-order system (9), then the Laplace transform $Y(s)$ of the output $y(t)$ is given by

$$Y(s) = \frac{\alpha_1 \left[ (s^2 K_D + s K_P + K_I) R(s) + s D(s) \right]}{s^2 (1 + \alpha_1 K_D) + s(\alpha_0 + \alpha_1 K_P) + \alpha_1 K_I} \tag{36}$$

where $D(s) = \mathcal{L}[d(t)]$. According to the Routh-Hurwithz stability criterion [12], the closed-loop system is stable if the gains $K_P$, $K_I$ and $K_D$ satisfy

$$K_P > -\frac{\alpha_0}{\alpha_1}, \qquad K_I > 0, \qquad K_D > -\frac{1}{\alpha_1} \tag{37}$$

It is well known that the effect of the constant disturbance $d_1$ is eliminated by means of the integral action of the controller. In order to verify this fact, two experiments are carried out with the CIT app and the filter (13). The fist one uses the parameters $K_P = x$, $K_I = 0$, and $K_D = x$. In the second experiment the PID controller employs the gains $K_P = x$, $K_I = 0$, and $K_D = x$. Figures x (a) and x (b) shows the fist and the second experiment, respectively. From these figures, it is evident that the response $y(t)$ of the second experiment is not affected by $d_1$.

### 8.1.2 Second order system

When the plant in Fig. Xx has the structure (14), the response $y(t)$ of the closed-loop system can be written in the Laplace domain as

$$Y(s) = \frac{\beta_2\left[(s^2K_D + sK_P + K_I)R(s) + sD(s)\right]}{s^3 + s^2(\beta_1 + \beta_2 K_D) + s(\beta_0 + \beta_2 K_P) + \beta_2 K_I} \tag{38}$$

For this closed-loop system the stability conditions are given by

$$(\beta_1 + \beta_2 K_D)(\beta_0 + \beta_2 K_P) > \beta_2 K_I, \qquad K_I > 0, \qquad K_D > -\frac{\beta_1}{\beta_2} \tag{39}$$

Figures xx (a) and xx (b) compares the signals $r(t)$ and $y(t)$ using $r(t) = 2.5H(t)$ and $r(t) = \sin(\omega t) + xx$, respectively. Note that in both cases $y(t)$ is close to $r(t)$.

## 8.2 Model reference adaptive controller (MRAC)

In comparison with the PID controller in that uses constant parameters, the MRAC has time-varying parameters [8], which are on-line adjusted by an adaptation law depending on the signals of the system to

be controlled (see Figure (Fig:BloackDiagramAdaptiveControl)). In order to design the MRAC, the structure of the system model is supposed to be known, but the nominal parameters of the system can be unknown. A reference model provides a desired response $y_m(t)$ to an input $r(t)$, and the adaptation law adjusts the parameters of a control law so that the response of the system $y(t)$ becomes the same as the reference model $y_m(t)$, or the tracking error error $e_m(t) = y(t) - y_m(t)$ becomes zero.

Next sections presents the adaptation laws of the MRAC obtained in absence of external disturbances, as shown in Fig. \ref{fig:marc}. MRAC controllers that are robust against disturbances are not considered in this paper, since they are beyond the scope of it; a detailed description of these controllers can be found in [9].

### 8.2.1 First order system

In this case the desired response of the system is provided by a reference model defined as

$$\dot{y}_m(t) + \alpha_{0m} y_m(t) = \alpha_{1m} r(t) \tag{40}$$

where $\alpha_{0m}$ and $\alpha_{1m}$ are positive constants.

The following control law, designed in [8] using the Lyapunov's direct method, guarantees that the tracking error $e_m(t)$ asymptotically converges to zero.

$$u(t) = \hat{\gamma}_1(t) r(t) + \hat{\gamma}_2(t) y(t) \tag{41}$$

where $y(t)$ is the response of the first-order system (9); moreover, the parameters $\hat{\gamma}_1(t)$ and $\hat{\gamma}_2(t)$ are adjusted using the following adaptation law

$$\dot{\hat{\gamma}}_1(t) = -\varrho e_m(t) r(t) \tag{42}$$

$$\dot{\hat{\gamma}}_2(t) = -\varrho e_m(t) y(t) \tag{43}$$

where $\varrho > 0$ is called adaptation gain.

In order to implement the MRAC on the CIT application, the expressions in (40)-(43) are discretized. The discretization version of the reference model is obtained using the zero-order hold method and it is given by

$$\begin{aligned} y_m(k) &= \sigma_0 y_m(k-1) + \sigma_1 r(k-1) \\ \sigma_0 &= e^{-\alpha_{0m} T_s}, \qquad \sigma_1 = \frac{\alpha_{1m}}{\alpha_{0m}}\left(1 - e^{-\alpha_{0m} T_s}\right) \end{aligned} \tag{44}$$

On the other hand, the trapezoidal numerical integration method [11] allows solving the adaptation laws (42) and (43). Thus, the discretization version of the MRAC controller is given by

$$u(k) = \hat{\gamma}_1(k) r(k) + \hat{\gamma}_2(k) y(k) \tag{45}$$

$$\hat{\gamma}_1(k) = \hat{\gamma}_1(k-1) - \frac{\varrho T_s}{2}\left[e_m(k) r(k) + e_m(k-1) r(k-1)\right] \tag{46}$$

$$\hat{\gamma}_2(k) = \hat{\gamma}_2(k-1) - \frac{\varrho T_s}{2}\left[e_m(k) y(k) + e_m(k-1) y(k-1)\right] \tag{47}$$

$$e_m(k) = y(k) - y_m(k) \tag{48}$$

The MRAC is applied to the low-pass filter in (13). Figure xx compares the responses $y(t)$ of this filter (\ref{e:rfs}) and the response $y_m(t)$ of a reference model with parameters $\alpha_{0m} = x$ and $\alpha_{1m} = x$, which is excited through the reference input $r = xx$. It is shown that the tracking error $e_m(t)$ is close to zero.

### 8.2.2 Second order system

In this case the reference model is defined as

$$\ddot{y}_m(t) + \alpha_{1m} \dot{y}_m(t) + \alpha_{0m} y_m(t) = \beta_{0m} r(t) \tag{49}$$

where $\alpha_{0m}$, $\alpha_{1m}$, and $\beta_{0m}$ are positive constants. The state-space representation of (49) is given as:

$$\dot{\mathbf{x}}_m(t) = \mathbf{A}_{mc}\mathbf{x}_m(t) + \mathbf{B}_{mc}r(t) \tag{50}$$

$$\mathbf{A}_{mc} = \begin{bmatrix} 0 & 1 \\ -\alpha_{0m} & -\alpha_{1m} \end{bmatrix}, \qquad \mathbf{B}_{mc} = \begin{bmatrix} 0 \\ \beta_{0m} \end{bmatrix}, \qquad \mathbf{x}_m(t) = [y_m(t),\, \dot{y}_m(t)]^T \tag{51}$$

The following MRAC controller guaranties that the error $e_m = y(t) - y_m(t)$ asymptotically converge to zero [9]:

$$u(t) = \mathbf{K}_c(t)\mathbf{x}(t) + \hat{L}(t)r(t) \tag{52}$$

$$\dot{\mathbf{K}}_c(t) = \gamma \mathbf{B}_{mc}^T \mathcal{P}\mathbf{E}(t)\mathbf{x}^T(t) \tag{53}$$

$$\dot{\hat{L}}(t) = \gamma \mathbf{B}_{mc}^T \mathcal{P}\mathbf{E}(t)r(t) \tag{54}$$

where $\gamma > 0$ is the adaptation gain, $\mathbf{x}(t) = [y(t),\, \dot{y}(t)]^T$, $\mathbf{E}_m(t) = [e_m(t),\, \dot{e}_m(t)]^T$ and matrix $\mathcal{P} = \mathcal{P}^T$ is the solution of the Lyapunv Equation

$$\mathbf{A}_{mc}^T \mathcal{P} + \mathcal{P}\mathbf{A}_{mc}^T = -\mathbf{I}_{2x2} \tag{55}$$

The CIT application computes the discretization versions of the reference model (50) and the MRAC (52) -(54), which are shown below. The digital implementation of reference model, produced with the zero-order method, is given as

$$\begin{gathered} \mathbf{x}_m(k+1) = \mathbf{A}_{md}\mathbf{x}_m(k) + \mathbf{B}_{md}r(k) \\ \mathbf{A}_{md} = \mathbf{e}^{\mathbf{A}_{mc}T_s}, \qquad \mathbf{B}_{md} = \left[\mathbf{A}_{md} - \mathbf{I}_{2x2}\right]\mathbf{B}_{mc}\mathbf{A}_{mc}^{-1} \end{gathered} \tag{56}$$

where the state transition matrix $\mathbf{e}^{\mathbf{A}_{mc}T_s}$ [7] is approximated as

$$e^{\mathbf{A}_{mc}T_s} = \mathbf{I}_{2x2} + \mathbf{A}_{mc}T_s + \frac{1}{2}\mathbf{A}_{mc}^2T_s^2 + \frac{1}{6}\mathbf{A}_{mc}^3T_s^3 + \frac{1}{24}\mathbf{A}_{mc}^4T_s^4 + \frac{1}{120}\mathbf{A}_{mc}^5T_s^5 + \frac{1}{720}\mathbf{A}_{mc}^6T_s^6 \tag{57}$$

On the other hand, the differential equations of the MRAC adaptation law are solved using the trapezoidal integration rule, thus obtaining

$$u(k) = \mathbf{K}_c(k)\mathbf{x}(k) + \hat{L}(t)r(k) \tag{58}$$

$$\mathbf{K}_c(k) == \mathbf{K}_c(k-1) + \frac{\varrho T_s}{2}\mathbf{B}_{mc}^T(k)\mathcal{P}\left[\mathbf{E}_m(k)\mathbf{x}^T(k) + \mathbf{E}_m(k-1)\mathbf{x}^T(k-1)\right] \tag{59}$$

$$\hat{L}(k) == \hat{L}(k-1) + \frac{\varrho T_s}{2}\mathbf{B}_{mc}^{T}(k)\mathcal{P}\left[\mathbf{E}_m(k)r(k) + \mathbf{E}_m(k-1)r(k-1)\right] \tag{60}$$

$$\mathbf{E}_m(k) = \mathbf{x}(k) - \mathbf{x}_m(k) \tag{61}$$

where $\mathbf{x}(k) = [y(k),\, \dot{y}(k)]^T$ . For computing the time derivative $\dot{y}(k)$, the following approximation is employed

$$\dot{y}(k) \approx \frac{y(k) - y(k-1)}{T_s} \tag{62}$$

An experiment is carried out with the CIT application, where the MRAC is applied to the second-order filter in (14), whose output $y(t)$ follows the response $y_m(t)$ of a reference model excited with the input $r(t) = xx$ (see Fig xx (a)). The parameters of this reference model are $\alpha_{0m} = x$ , $\alpha_{1m} = x$ , and $\beta_{0m} = x$ . From Fig. Xx (b), it is possible to see that the tracking error $e_m(t)$ converges in a neighborhood around zero.

# 9 Implementation in the classroom

This app is introduced to the bachelors students of UAC y UAdeC. The students shown interest in interacting with the connections and it made the classroom more interactive and the students more engaged with the classroom activity. Few students who were unable to access OTG through their phone used their laptop to install virtual box with android-x86 to install this application \cite{Androidx86}.

# 10 Conclusions

# 11 References


1. Åström KJ (Karl J, Hägglund T, Åström KJ (Karl J. *PID Controllers*. International Society for Measurement and Control; 1995. https://ww2.isa.org/store/products/product-detail/?productId=116103. Accessed June 10, 2018.

2. Norris C, Hossain A, Soloway E, Thurnau AF. Using Smartphones as Essential Tools for Learning A Call to Place Schools on the Right Side of the 21st Century. http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.456.3319&rep=rep1&type=pdf. Accessed June 10, 2018.

3. Gadi SK. ControlToolbox. June 2018. doi:10.5281/ZENODO.1291275

4. Gadi SK. Control Toolbox (Time domain) – Apps on Google Play. https://play.google.com/store/apps/details?id=com.skgadi.controltoolboxtimedomain. Accessed June 11, 2018.

5. Android Graph View plotting library. http://www.android-graphview.org/. Accessed May 31, 2018.

6. AsyncTask | Android Developers. https://developer.android.com/reference/android/os/AsyncTask. Accessed June 12, 2018.

7. Ogata K. *Modern Control Engineering*. Prentice-Hall; 2010.

8. Slotine J-JE (Jean-JE., Li W. *Applied Nonlinear Control*. Prentice Hall; 1991.

9. Ioannou PA (Petros A., Sun J. *Robust Adaptive Control*. https://books.google.com.mx/books?id=ffavAAAAQBAJ&dq=P.+A.+Ioannou+and+J.+Sun&source=gbs_navlinks_s. Accessed June 12, 2018.

10. Shieh LS, Wang H, Yates RE. Discrete-continuous model conversion. *Appl Math Model*. 1980;4(6):449-455. doi:10.1016/0307-904X(80)90177-8

11. Buso S, Mattavelli P. *Digital Control in Power Electronics*. Morgan & Claypool Publishers; 2006. https://books.google.com.mx/books?id=yTc-yzAaRCQC&dq=Digital+Control+in+Power+Electronics&source=gbs_navlinks_s. Accessed June 12, 2018.

12. Golnaraghi MF, Kuo BC. *Automatic Control Systems*. https://books.google.com.mx/books?id=TMHRjwEACAAJ&dq=Automatic+Control+Systems+golnaraghi&hl=en&sa=X&ved=0ahUKEwiR25z40s_bAhUHXKwKHfYBDesQ6AEIKTAA. Accessed June 12, 2018.