Congreso Internacional en Ingeniería Electrónica. Mem. Electro 2015, Vol. 37, pp 164-169, Chihuahua, Chih. México
http://depi.itchihuahua.edu.mx/display/memorias_electro/MemoriaElectro2015
ISSN 1405-2172

POLE PLACEMENT CONTROL OF A TIME-DISCRETIZED SYSTEM IN A STATE-SPACE IMPLEMENTATION WITH MATLAB AND ARDUINO

David Sáenz-Zamarrón[1], Sergio R. Muñoz-Sandoval, Noé M. Ponce-Domínguez, Nancy Ivette Arana-de-las-Casas, Mario I. Chacón-Murguía, J. Francisco Alatorre-Ávila
[1]Metal-Mechanics Department, Technological Institute of Cuauhtémoc City.davsaenz@gmail.com,

RESUMEN

En este trabajo se describe el diseño del control para un sistema dinámico SISO discretizado utilizando Matlab®, Simulink® y el Arduino Due. Se creó una plataforma que puede ser usada con fines educativos y de investigación. Se describen los componentes de hardware, software y control. Se presenta la forma en que se implementó el algoritmo de control corriendo en un sistema digital embebido. La variable a controlar es medida con osciloscopio y se discuten los compromisos entre la constante de tiempo del sistema con la tasa de muestreo. El diseño del sistema de control esta escrito en código de Matlab, Simulink y está implementado en el Arduino Due. Se presenta el modelo matemático y se analiza el comportamiento de la respuesta temporal y de estado estable. La aportación del proyecto es la posibilidad de alcanzar un mayor entendimiento del comportamiento del algoritmo de control e implementarlo en una infraestructura digital.

Palabras Clave: Control digital, espacio de estados, Arduino Due.

ABSTRACT

This paper describes the control design for a discretized dynamic system using Matlab®, Simulink®, and Arduino Due. A flexible platform was created that can be used for educational and research purposes. Hardware, software and control components are described throughout this document. The way in which the control algorithm runs in an embedded digital system is shown. The variable to be controlled is measured with an oscilloscope and the commitments between system's time constant and sampling rate are discussed. Control system design is written in Matlab code & Simulink and is deployed in Arduino Due. The mathematical model is presented, and temporal response behavior is analyzed. The main contribution of this project is the achieving of a greater understanding of control algorithms and the possibility of implement them in a digital infrastructure.

Keywords: Digital control, space state, Arduino Due.

1. INTRODUCCION

Control engineering is a multidisciplinary area of knowledge in which systems require to comply with performance specifications such as speed of reaction of a variable of interest. In recent years due to the availability of low-cost digital computers the use of digital controllers has increased in control systems. The flexibility in control programs is the main advantage of digital control systems.

In digital control systems their dynamics can be described through a difference equation that depends on discrete time *k*, when numerical values of all its coefficients are provided.

State space representation implies a mathematical model of an *n* order discretized dynamic system described through a set of inputs, states variables and outputs related by *n* first-order difference equations which are chained to form a matrix structure.

Pole assignment control technique, among others reported in [1-7], requires the closed-loop transfer function poles allocation to a desired location that will meet some design requirement, but most of other works do not dedicate too much effort to implement algorithms on a digital platform.

This article describes the design procedure and report experiences gained during its construction and implementation. It contributes mainly with a detailed description of the system design and how a very accessible platform as the Arduino Due is used to implement real-time control algorithms.

Analysis was assisted using mathematical software, electronic circuit simulators, analog and digital computers; measurements were done in oscilloscope to verify mathematical results.

Digital controller design experimentation was done through pole placement technique with several discretized custom analog plants designed by using operational amplifiers to generate an oscilloscope easy-to-see response. Dynamic systems are discretized under state space model, a desired response specification is defined, and a controller was designed with Matlab & Simulink and was implemented in a digital system such as the Arduino Due.

Plants are defined as first and second order; they were characterized by their step input test response as time constant, overshoot and settling time.

The original response is obtained by injecting a step signal into a space state represented system, then the system response is tested with an analytically designed compensator by using Matlab and Simulink, then a code for Arduino Due was written to observe its real-time behavior in an oscilloscope.

After the discrete pole placement controller was designed, it also was modified to be able to precisely follow the input, and was programmed in the Arduino Due development platform.

2. SYSTEM MODEL

Three illustrative dynamic systems were created, two of them were second order and the other was the first order one, in table 1 the dynamic systems specifications are shown.

Table 1. The dynamic systems specifications

System	Original Specification	Wanted specification
S1 2nd Order	$t_s = 5\ ms$ $Mp = 40\%$	$t_s = 5\ ms$ $Mp = 24\%$
S2 2nd Order	$t_s = 1.6ms$ M_p=45%	$t_s = 1ms$ M_p=20%
S3 1ft Order	$\tau = 1ms$ e_{ss}=50%	$t_s = 2.2ms$ M_p=16% e_{ss}=0

Following is described the way in which the S1 system was design, the others were made similarly.

2.1 Determination of the S1 system plant

A 2nd order analog plant containing the required specifications proposed in table 1 was design.

At the beginning it must have 5 milliseconds of settling time and 40% of overshoot as shown in equations (1) and (2)

$$t_s = 0.005 \quad (1)$$

$$Mp = 40\% \quad (2)$$

Later, 2nd order closed loop system is represented in the form of equation (3)

$$\frac{C(s)}{R(s)} = \frac{{\omega_n}^2}{s^2+2\xi\omega_n s+{\omega_n}^2} \quad (3)$$

This type of system has a wide variety of responses depending on parameters: ξ, the damping ratio and ω_n, the natural frequency.

The system's characteristic equation is shown in equation (4)

$$s^2 + 2\xi\omega_n s + {\omega_n}^2 = 0 \quad (4)$$

To determine the damping ratio and the natural frequency, which characterize this particular second order system, a series of mathematical calculations were done.

First the constant time τ is determined throughout the t_s value, and then σ is obtained.

$$\tau = \frac{t_s}{4} = \frac{5ms}{4} = 1.2\ ms \quad (5)$$

$$\sigma = \frac{1}{\tau} = 833.333 \quad (6)$$

The damped frequency ω_d is a function of the overshoot and is given by equation (7)

$$\omega_d = \frac{-\sigma}{\ln\left(\frac{Mp}{100}\right)}\ \pi = \frac{-833.33}{\ln\left(\frac{40}{100}\right)}\ \pi\ = 2857.1536 \quad (7)$$

Angle β is determined by equation (8) as follows

$$\beta = tg^{-1}\left(\frac{\omega_d}{\sigma}\right) = tg^{-1}\left(\frac{2857.1536}{833.33}\right) = 73.7399^o \quad (8)$$

Later the value of the damping ratio ξ is determined

$$\xi = \cos\beta = \cos\ (73.74°) = 0.2799 \quad (9)$$

Following the natural frequency is calculated.

$$\omega_n = \frac{\omega_d}{\sqrt{1-\xi^2}} = \frac{2857.1536}{\sqrt{1-0.2799^2}} = 2976.1112 \quad (10)$$

After all values are substituted to reach a plant transfer function with a unit feedback as shown in equation (11)

$$G_p(s) = \frac{{\omega_n}^2}{s(s+2\xi\omega_n)} = \frac{(2976)^2}{s(s+1666)} \quad (11)$$

2.2 Plant electronic design

Following the procedure performed to determine the value of electronic components to be used for voltage transient response as specified by the S1 system against a square signal by using operational amplifiers is shown.

To start equation (11) breaks down so that it is expressed as the product of a pole at the origin and another outside the origin as in equation (12)

$$Gp(s) = \frac{(2976)^2}{s^2+1666s} = \frac{(2976)^2}{s} * \frac{1}{s+1666} \quad (12)$$

Equation (13) is used as the part of pole at origin, this is an integration operation.

$$\frac{C_1(s)}{E_1(s)} = \frac{\frac{1}{C_1}}{\frac{R_1}{1}} = \frac{1}{R_1C_1s} \quad (13)$$

The stage of pole outside origin is obtained with equation (14)

$$\frac{C_2(s)}{C_1(s)} = \frac{\frac{R_3}{R3C2+1}}{\frac{R_2}{1}} = \frac{R_3}{R_2(R_3C_2+1)} \quad (14)$$

After a series of algebraic manipulations, the plant's resistors and capacitors are shown in Figure 1. Full analog control feedback system is also observed.

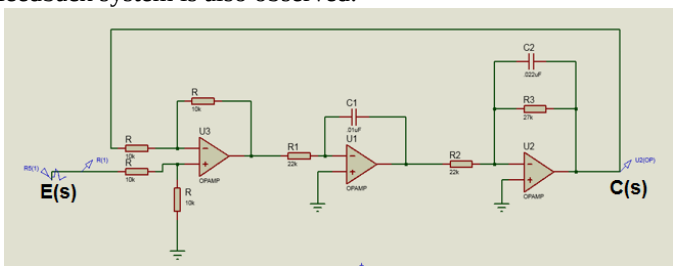

Figure 1. Plant with unit feedback

Figure 2 shows the response obtained with Proteus simulation against step input.

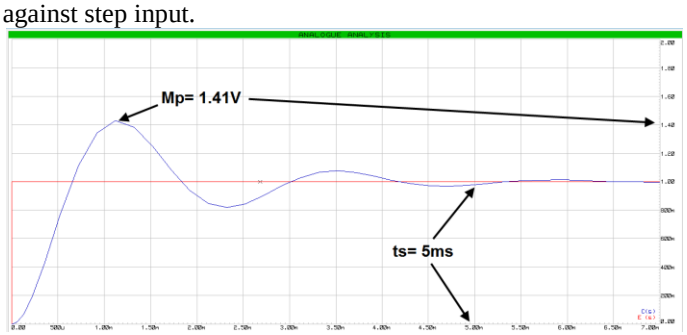

Figure 2 Response of plant to step input in Proteus

Shown in figure 2 is the response (in blue) for 1V step input (in red), it has an 1.41V overshoot, meaning 41% more than the input value, very close to the 40% used for the plant design seen in equation (2) as well as the settling time that, according to equation (1), it shall be 5 msec, which in figure 2 is observed that is the precise time when the system started to stabilize.

Subsequently an electronic circuit was made, to which a train of square pulses were applied as the step input. In figure 3 and 4 the measurement carried out in the oscilloscope with an overshoot that can be seen, it is slightly more than 40% higher than the input signal and a settling time of 5ms.

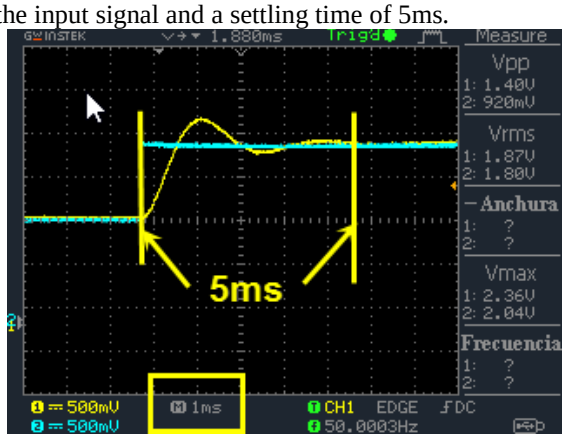

Figure 3 Settling time in phisical plant.

ISSN 1405-2172

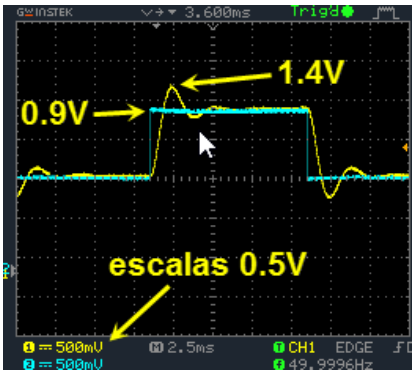

Figure 4 Overshoot in oscilloscope

Finally table 2 shows the gotten temporary specifications and the transfer functions for the three designed analog systems.

Table 2. Dynamic system specifications.

System	specification	$\frac{C(s)}{R(s)}$
S1	$t_s = 5\ ms$ $Mp = 40\%$	$\frac{(2976)^2}{s^2 + 1666s + 2976^2}$
S2	$t_s = 1.6ms$ M_p=45%	$\frac{103x10^6}{s^2 + 5000s + 103x10^6}$
S3	$\tau = 1ms$ e_{ss}=50%	$\frac{1000}{s + 2000}$

It´s worth to mentioning that for the S3 system it was necessary to include an extra pole with transfer function $\frac{10000}{s+10000}$ in order to make S3 a 2nd order system without changing its original characteristics.

3. SYSTEM DISCRETIZATION

The Arduino Due is where the digital control system will be implemented, making operations of sampler & data-hold (ADC & DAC), it will perform difference equations which are the discrete version of differential equations of the analog version.

3.1 State space

Next, the discretization of the analog system of equation (11) in state space form is shown.

First it begins from the plant's transfer function

$$\mathrm{Gp(s)} = \frac{2976^2}{\mathrm{s(s+1666)}} = \frac{\mathrm{Y(s)}}{\mathrm{U(s)}} \quad (15)$$

The continuous state equations are found.

$$\mathcal{L}^{-1}\{2976^2\mathrm{U(s)} = \mathrm{s^2Y(s)} + 1666\mathrm{sY(s)}\} \quad (16)$$

$$\ddot{y}(t) = -1666\dot{y}(t) + 2976^2 u(t)$$

The phase variables x_1 and x_2 are defined

$$x_1(t) = y(t) \quad (17)$$

$$\dot{x}_1(t) = \dot{y}(t)$$

$$x_2(t) = \dot{x}_1(t) = \dot{y}(t) \quad (18)$$

$$\dot{x}_2(t) = \ddot{x}_1(t) = \ddot{y}(t)$$

$$\dot{x}_1(t) = 0x_1(t) + 1 \quad x_2(t) - 0 \quad u(t) \quad (19)$$

$$\dot{x}_2(t) = 0x_1(t) - 1666x_2(t) + 2976^2 u(t) \quad (20)$$

$$y(t) = 1x_1(t) + 0 \quad x_2(t) \quad (21)$$

With this, it is obtained the continuous state equations in matrix form with phase variables [4].

$$\begin{bmatrix}\dot{x}_1\\ \dot{x}_2\end{bmatrix} = \begin{bmatrix}0 & 1\\ 0 & -1666\end{bmatrix}\begin{bmatrix}x_1\\ x_2\end{bmatrix} + \begin{bmatrix}0\\ 2976^2\end{bmatrix}u \quad y = [1 \quad 0]\begin{bmatrix}x_1\\ x_2\end{bmatrix} \quad (22)$$

Where $\begin{bmatrix}0 & 1\\ 0 & -1666\end{bmatrix}$ is $\mathbf{A}_c$ and $\begin{bmatrix}0\\ 2976^2\end{bmatrix}$ is $\mathbf{B}_c$

The continuous-time state-transition matrix is obtained [5,6].

$$\mathbf{\Phi}_C(t) = \mathcal{L}^{-1}\{[s\mathbf{I} - \mathbf{A}_C]^{-1}\} \quad (23)$$

$$= \mathcal{L}^{-1}\left\{\left[\begin{bmatrix}s & 0\\ 0 & s\end{bmatrix} - \begin{bmatrix}0 & 1\\ 0 & -1666\end{bmatrix}\right]^{-1}\right\}$$

$$= \mathcal{L}^{-1}\left\{\left[\begin{bmatrix}s & -1\\ 0 & s+1666\end{bmatrix}\right]^{-1}\right\}$$

$$= \mathcal{L}^{-1}\left\{\frac{1}{s(s+1666)-0}\begin{bmatrix}s+1666 & 1\\ 0 & s\end{bmatrix}\right\}$$

$$= \mathcal{L}^{-1}\left\{\begin{bmatrix}\frac{1}{s} & \frac{1}{s(s+1666)}\\ 0 & \frac{1}{(s+1666)}\end{bmatrix}\right\}$$

$$= \mathcal{L}^{-1}\left\{\begin{bmatrix}\frac{1}{s} & \frac{1}{1666s} - \frac{1}{1666(s+1666)}\\ 0 & \frac{1}{(s+1666)}\end{bmatrix}\right\}$$

$$= \begin{bmatrix}1 & \frac{1}{1666} - \frac{e^{-1666t}}{1666}\\ 0 & e^{-1666t}\end{bmatrix} \quad (24)$$

Arrays **A** and **B** are obtained, that is, discrete state equations for the sampled-data system. For **A**, t is replace by T=0.00012, the sampling period used, which is 11.6 times smaller than the time constant of the system (τ), fulfilling sampling theorem.

$$\mathbf{A} = \mathbf{\Phi}_C(T)|_{T=.00012} = \left.\begin{bmatrix}1 & \frac{1}{1666} - \frac{e^{-1666T}}{1666}\\ 0 & e^{-1666T}\end{bmatrix}\right|_{T=.00012} = \begin{bmatrix}1 & 0.000108766\\ 0 & 0.8187963\end{bmatrix} \quad (25)$$

To obtain the discreet array **B** it is integrated from 0 to T with respect to τ.

$$\mathbf{B} = \left[\int_0^T \Phi_C(\tau)\,d\tau\right]B_C = \begin{bmatrix}\int_0^T 1\,d\tau & \int_0^T \left(\frac{1}{1666} + \frac{e^{-1666\tau}}{2775556}\right)d\tau\\ 0 & \int_0^T \left(\frac{e^{-1666\tau}}{1666}\right)d\tau\end{bmatrix}\begin{bmatrix}0\\ 2976^2\end{bmatrix} \quad (26)$$

$$\mathbf{B} = \begin{bmatrix}0.00012 & 6.74283x10^{-9}\\ 0 & 0.000108766\end{bmatrix}\begin{bmatrix}0\\ 2976^2\end{bmatrix} = \begin{bmatrix}0.0597184\\ 963.294\end{bmatrix} \quad (27)$$

With the same continuous and discrete matrices $\mathbf{C} = \mathbf{C}_c$ $\mathbf{D} = \mathbf{D}_c$

With this, discrete state equations are obtained

$$\mathbf{x}(k+1) = \begin{bmatrix}1 & 0.000108766\\ 0 & 0.8187963\end{bmatrix}\mathbf{x}(k) + \begin{bmatrix}0.0597184\\ 963.294\end{bmatrix}u(k)$$

$$y(k) = [1 \quad 0]\mathbf{x}(k) \quad (28)$$

Now it is wished to make a control by pole placement for the prior representation with the following equation.

$$\mathbf{x}(k+1) = (\mathbf{A} - \mathbf{Bk})\mathbf{x}(k) + \mathbf{B}u(k) \qquad y(k) = [1 \quad 0]\mathbf{x}(k) \quad (29)$$

First $\mathbf{A}_p$ is obtained

$$\mathbf{A}_p = (\mathbf{A} - \mathbf{Bk}) = \begin{bmatrix}1 & 0.000108766\\ 0 & 0.8187963\end{bmatrix} - \begin{bmatrix}0.0597184\\ 963.294\end{bmatrix}[k_1 \quad k_2]$$

$$= \begin{bmatrix}1 - 0.0597184k_1 & 0.000108766 - 0.0597184k_2\\ -963.294k_1 & 0.8187963 - 963.294k_2\end{bmatrix} \quad (30)$$

Next the characteristic equation is obtained as a function of k_1 and k_2 by using the determinant of the rest of z by the identity matrix with the A_p matrix obtained.

$$|z\mathbf{I} - \mathbf{A}_p| = 0 \quad (31)$$

$$\left|\begin{bmatrix}z & 0\\ 0 & z\end{bmatrix} - \begin{bmatrix}1 - 0.0597184k_1 & 0.000108766 - 0.0597184k_2\\ -963.294k_1 & 0.8187963 - 963.294k_2\end{bmatrix}\right| = 0 \quad (32)$$

$$z^2 + (0.0597184k_1 + 963.294k_2 - 1.8188)z + 0.0558764k_1 - 963.294k_2 + 0.818796 = 0 \quad (33)$$

Values 1 and 0 for k_1 and k_2 are assumed respectively to produce a unit feedback.

$$k_1 = 1,\ k_2 = 0 \quad (34)$$

To obtain the following state equation

$$\mathbf{x}(k+1) = \begin{bmatrix}0.9403 & 0.000108766\\ -963.2940 & 0.8187963\end{bmatrix}\mathbf{x}(k) + \begin{bmatrix}0.0597184\\ 963.294\end{bmatrix}u(k) \quad (35)$$

Figure 5 shows the comparison between the response of the analog system and its discrete version.

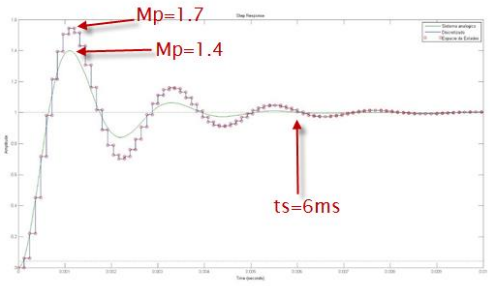

Figure 5 Analog vs. discrete

3.2 Pole placement

It is desired to reduce M_p=24% and t_s=4.8msec so, it's required to have

$$\xi_{deseado} = 0.4121 \qquad \tau_{deseado} = 0.0014 \qquad (36)$$

The new pole ratio is calculated for the τ wished.

$$r = e^{-T/\tau} \quad r = e^{-0.00012/0.0014} = 0.917864 \qquad (37)$$

The new pole angle is calculated for the wished ξ and τ

$$\theta = \sqrt{\frac{\ln^2 r}{\xi} - \ln^2 r} \qquad \theta = \sqrt{\frac{\ln^2 0.917864}{0.4121} - \ln^2 0.917864} = 0.189376 \quad (38)$$

Once the desired complex conjugate poles are obtained $z_{1,2}$ in polar format it is transformed to its rectangular format.

$$z_{1,2} = 0.917864\angle \pm 0.189376\text{rad} = 0.901456 \pm 0.172777i \qquad (39)$$

Then the desired poles become a characteristic equation.

$$z^2 - 1.80291z + 0.842475 = 0 \qquad (40)$$

It is resolved by the direct method, substituting the desired values of equation (40) in equation (33) to obtain the values of k_1 and k_2 [6].

$$0.0597184k_1 + 963.294k_2 - 1.8188 = -1.80291$$
$$0.0558764k_1 - 963.294k_2 + 0.818796 = 0.842475$$
$$k_1 = 0.342308 \qquad k_2 = -4.72552 * 10^{-6} \qquad (41)$$

The system is built in Simulink with these values as shown in Figure 6.

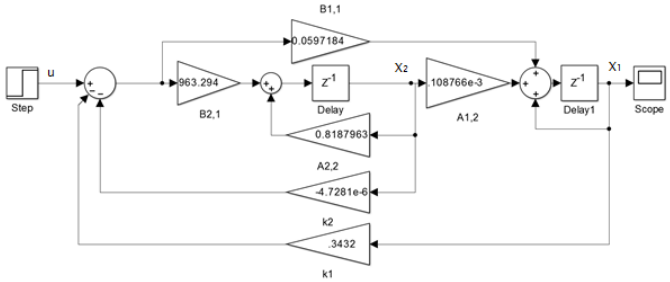

Figure 6 Discrete space state block diagram

Well known is the effect on the steady state error when using pole placement controller, so it will be eliminated by scheme of equation (42) which allow it to follow the input [6,7].

$$\begin{bmatrix} \mathbf{A}-\mathbf{I} & \mathbf{B} \\ \mathbf{C} & 0 \end{bmatrix}\begin{bmatrix} \mathbf{G} \\ \mathbf{G_p} \end{bmatrix} = \begin{bmatrix} 0 \\ \mathbf{I} \end{bmatrix} \qquad (42)$$

$$\left(\begin{bmatrix} 1 & 0.000108766 \\ 0 & 0.8187963 \\ & [1 \quad 0] \end{bmatrix} - \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}\right) \begin{bmatrix} 0.0597184 \\ 963.294 \\ 0 \end{bmatrix}\begin{bmatrix} G_1 \\ G_2 \\ G_p \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \qquad (43)$$

$$\begin{bmatrix} 0 & 0.000108766 & 0.0597184 \\ 0 & -0.1812037 & 963.294 \\ 1 & 0 & 0 \end{bmatrix}\begin{bmatrix} G_1 \\ G_2 \\ G_p \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \qquad (44)$$

Finding $\begin{bmatrix} G_1 \\ G_2 \\ G_p \end{bmatrix}$ inverting the first array

$$\begin{bmatrix} G_1 \\ G_2 \\ G_p \end{bmatrix} = \begin{bmatrix} 0 & 0.000108766 & 0.0597184 \\ 0 & -0.1812037 & 963.294 \\ 1 & 0 & 0 \end{bmatrix}^{-1}\begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} =$$

$$\begin{bmatrix} 0 & 0 & 1 \\ 8333037 & -0.516618 & 0 \\ 1.56758 & 0 & 0 \end{bmatrix}\begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \qquad (45)$$

With this, **G** and G_p are obtained

$$\mathbf{G} = \begin{bmatrix} G_1 \\ G_2 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \qquad G_p = 0 \qquad (46)$$

Gains producing zero steady-state error are obtained and replaced in the next state equation

$$\mathbf{x}(k+1) = [\mathbf{A} - \mathbf{Bk}]\mathbf{x}(k) + \mathbf{B}[\mathrm{Gp} + \mathbf{kG}]u(k) \qquad (47)$$

$$\mathbf{A} - \mathbf{Bk} =$$

$$\begin{bmatrix} 1 - 0.0597184(0.342308) & 0.000108766 - 0.0597184(-4.72552 * 10^{-6}) \\ -963.294(0.342308) & 0.8187963 - 963.294(-4.72552 * 10^{-6}) \end{bmatrix} =$$

$$\mathbf{A} - \mathbf{Bk} = \begin{bmatrix} 0.979558 & 0.000109048 \\ -329.743 & 0.823348 \end{bmatrix} \qquad (48)$$

$$\mathbf{B}[\mathrm{Gp} + \mathbf{kG}] = \begin{bmatrix} 0.0597184 \\ 963.294 \end{bmatrix}\left[0 + [0.342308 \quad -4.72552 * 10^{-6}]\begin{bmatrix} 1 \\ 0 \end{bmatrix}\right] \qquad (49)$$

$$\mathbf{B}[\mathrm{Gp} + \mathbf{kG}] = \begin{bmatrix} 0.0597184 \\ 963.294 \end{bmatrix}[0 + 0.342308] \qquad (70)$$

$$\mathbf{x}(k+1) = \begin{bmatrix} 0.979558 & 0.000109048 \\ -329.743 & 0.823348 \end{bmatrix}\mathbf{x}(k) + \begin{bmatrix} 0.0204421 \\ 329.743 \end{bmatrix}u(k)$$
$$y(k) = [1 \quad 0]\mathbf{x}(k) \qquad (71)$$

The system is built in Simulink with these **G** and G_p values as shown in Figure 7.

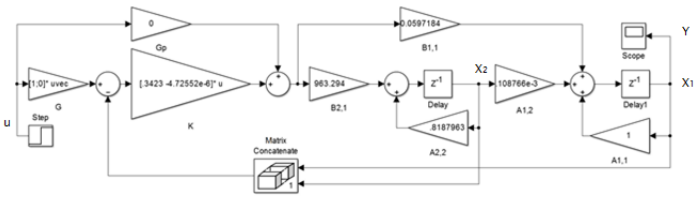

Figure 7 System with **G** and G_p values

In figure 8 Simulink output is shown with the steady-state error corrected.

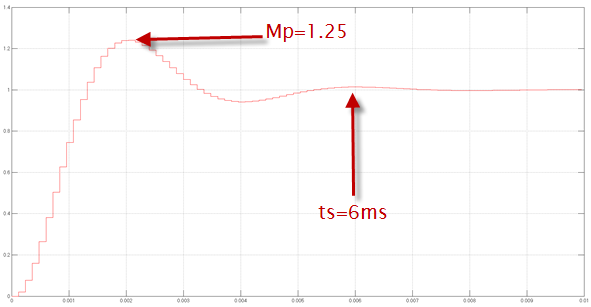

Figure 8 Simulink response

In figure 9 the comparison between the system without control (black), with the settling time and overshoot corrected (red) and with the steady-state error corrected (blue).

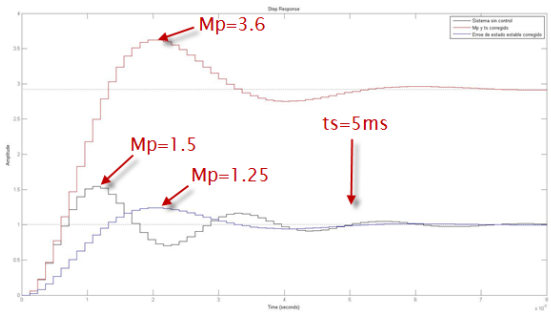

Figure 9 Compensated and uncompensated response

3.3 Arduino program

The Arduino Due has 12 bits resolution in ADCs and DACs, DueTimer library with 8 temporization channels and integer and floating point number processing capacity of, that allow it to implement the sampler & zero-order-hold of the data acquisition system, also the comparator and compensator.

Below is the code used in the Arduino program to perform the controller by the pole placement procedure and the internal plant also build into the microprocessor.

In the following line the library "DueTimer.h" is imported, which allows the user to use the Arduino Due timers.

```
#include <DueTimer.h>
```

Port A0 is declared in the Analogico0 variable, which will be used later to make ADC conversions.

```
const int Analogico0=0;
```

Variable declaration of usec contains time unit in microseconds.

```
const int usec=1000000;
```

Square signal generator parameters are declared.

```
volatile int amplitude=1024;
volatile int offset=1250;
volatile bool flag=false;
```

In the following lines the discrete time *k* and the input *r* are declared.

```
int k;
volatile int r;
```

Statement of the times that will be used in the timers *Tr* (1/2 of the period of the input signal *r*) and *Td* (sampling period)

```
int Tr=.01*usec;
int Td=.00012*usec;
```

In the following lines the coefficients of plant state equation are declared, as well as gains K, G1, G2 and Gp that will be the values that control the plant.

```
volatile float u=0;
volatile float x1=0,x2=0;
volatile float A[N][N]={ 1,0.108766E-3,0,0.8187963 };
volatile float B[N][1]={ 0.0597184,963.294 };
volatile float C[1][N]={1,0};
volatile float D=0;
volatile float y=0;
```

Gain declarations of K, G1, G2 and Gp.

```
volatile float K[1][N]={0.342308,-4.72552E-6 };
volatile float G[N][1]={1,0};
volatile float Gp=0;
```

The following subroutine generates a square wave signal (step input) with amplitude declared previously.

```
void SetPoint()
{
  if (!flag) r=-amplitude;
  else r=amplitude;
    analogWrite(DAC0,r+offset);
  flag=!flag;
}
```

In the following subroutine all the plant operations are done and the controller through states equations.

```
void Dynamic()
{
```

It is stored in the input variable *u* the result of control operations.

```
  u=Gp*r+K[0][0]*(G[0][0]*r-x1)+K[0][1]*(G[1][0]*r-x2);
```

Calculation of state variable value x_1

```
  x1= A[0][0]*x1 + A[0][1]*x2 + B[0][0]*u;
```

Calculation of state variable value x_2

```
  x2= A[1][0]*x1 + A[1][1]*x2 + B[1][0]*u;
```

Output value calculation

```
  y = C[0][0]*x1 + C[0][1]*x2;
```

Through DAC1 the controlled plant result is sent out (*y* value).

```
  analogWrite(DAC1,int(y)+offset);}
```

In the next subroutine the parameters required to execute the program are initialized.

```
void setup()
{
```

DAC0 and DAC1 are declared as outputs and Analogico0 as input.

```
  pinMode(DAC0,OUTPUT);
  pinMode(DAC1,OUTPUT);
  pinMode(Analogico0,INPUT);
```

DACs and ADCs are assigned with 12 bits of resolution.

```
  analogReadResolution(12);
  analogWriteResolution(12);
  analogReference(eAnalogReference(DEFAULT));
```

In the following lines the subroutine that is going to be working with each timer will be assigned, as well as the time corresponding to each one.

```
  Timer4.attachInterrupt(SetPoint);
  Timer5.attachInterrupt(Dynamic);
  Timer4.start(Tr);
  Timer5.start(Td);
}
```

This subroutine will be executed while the program is running.

```
void loop()
{
  delay(200);
}
```

Figure 10 shows the capture in oscilloscope for the zero steady-state error pole placement compensator; there it can be seen the performance of compensated plant.

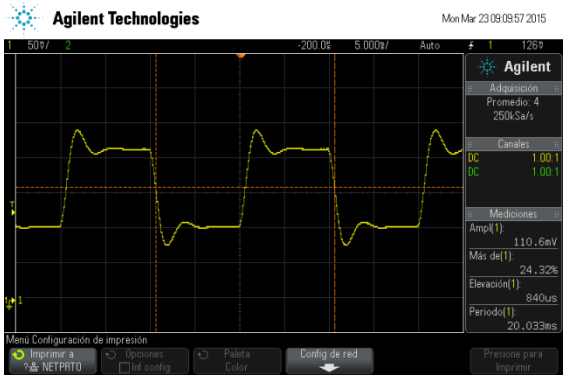

Figure 10 Oscilloscope view of controlled plant.

4. RESULTS

Below are the results obtained with three different dynamic systems S1, S2 and S3 which the control was applied to. The controller was carried out in order to improve the response of the original system, controller coefficients were obtained in Matlab & Simulink and were used in the Arduino. Table 3 shows the results obtained in order to compare the values of each one of them.

Table 3 Results in Matlab® and Arduino

SYSTEM	MATLAB®		ARDUINO	
	MP%	ts(mseg)	MP%	ts(mseg)
S1	24	4.92	24	4.92
S2	20	0.95	22	1
S3	16	2.2	16	2.2

5. CONCLUSIONS

A design and implementation of a control scheme was developed in which a variety of tools were used: mathematics, electronics and programming in order to make that the mathematical function containing the plant complies with the desired specification thanks to the designated controller gains. This work involved the construction of three OpAmps circuits, as well as codes in Matlab, Simulink and Arduino.

6. BIBLIOGRAPHY

[1] Sarawut S., Witchupong W. (2011). “State-PID Feedback for Pole Placement of LTI Systems”. Mathematical Problems in Engineering. Hindawi Publishing Co. vol. 2011.

[2] Abdelaziz T.H.S., and Valasek M. (2004). “Pole-placement for SISO linear systems by state-derivative feedback,” *IEEE Proceedings: Control Theory and Applications*, vol. 151, no. 4, pp. 377–385.

[3] García G., Tarbouriech S., Gomes Da Silva J.M., Castelan E.B. (2004). Pole assignment in a disk for linear systems by static output feedback. IEE Proc. Control Theory Appl. Vol. 151, No. 6, November 2004.

[4] Nise N.S. (2007). Control System Engineering. Ed. Wiley, 5ta Edition. USA. ISBN: 978-0471-79475-2.

[5] Phillips C.L., Nagle H.T. (1995). Digital Control System Analysis and Design. Ed. Prentice Hall 3rd. Edition. New Jersey, USA. ISBN: 0-13-309832-X.

[6] Fernández del Busto y Ezeta R. (2013). Análisis y Diseño de Sistemas de Control Digital. McGraw Hill Education. ISBN: 978-607-15-0773-0.

[7] Franklin G.E., Powell J.D., Workman M.L. (1998). Digital Control of Dynamic Systems. 3rd. Ed. Menlo Park, CA. Addison-Wesley Longman, Inc.