14

MATLAB as Interface for Intelligent Digital Control of Dynamic Systems

João Viana da Fonseca Neto and Gustavo Araújo de Andrade
Federal University of Maranhão
Department of Electrical Engineering
Control Process Laboratory
Brazil

1. Introduction

Digital control systems have shown to the developers the features and their applications in a wide variety of plants. Tools that help the developer to promote a design methodology that is efficient and at the same time reliable, has gained ground in the market and attributions of control engineers. Tasks such as data acquisition system, control design and system implementation can become arduous if there are prerequisites for sensitivity and complexity of the controller for these tasks take time and performance of the developer which will entail additional costs to the final product. These tools provide the designer with the scientist and the smooth progress of their work role has important and necessary in many areas will be subject to this study. The integration of technologies to speed and lower costs as it relates the design phases can be observed with a proper exploration of the work tool and how knowledge of plant and control techniques that meet in a less costly the goal of being achieved whatever the difficulties of the project. The choice of a tool properly can be a deciding factor in a world where time and efficiency of processes is become extremely important because the applications are growing in scale and more complex, Moudgalya (2007), Andrade (2010).
Intelligent Systems has considerable performance in accordance with plans and small large and its design encompasses more robustness to the system as well as ease of expansion. The mathematical simplicity that fuzzy systems can present and adaptability of neural networks are adopted more frequently in the academy and the industry.
An approach of intelligent systems requires a systematic and efficient operation because one works with a data stream that needs a consistency so that the iterations will be where made all the decisions may have minimal accounting. A tool that can assist in the design of this requirement is most welcome in engineering projects for control system parameter becomes constant with time,S. Sumathi (2010).
The *MATLAB* software designers can provide facilities for development and interface with different technologies for data acquisition through its communication protocols such as Serial Interface, OPC, Ethernet and others. This work is mainly focused on a methodology design of digital control systems using as development platform and implementing in software *MATLAB*.

2. Digital control and Data Acquisition Systems

The Data Acquisition System (DAS)is a key part to the project and implementation of digital controllers for the nature of digital systems is based on sampled analog system, ie played by the analog system a digital computer and its Since the interface Digital - Analog system performs the control actions . The Figure 1 shows a diagram of a system digital control and its special features such as the pool Analog - Digital and Digital - Analogue, responsible for interfacing with the analog environment.

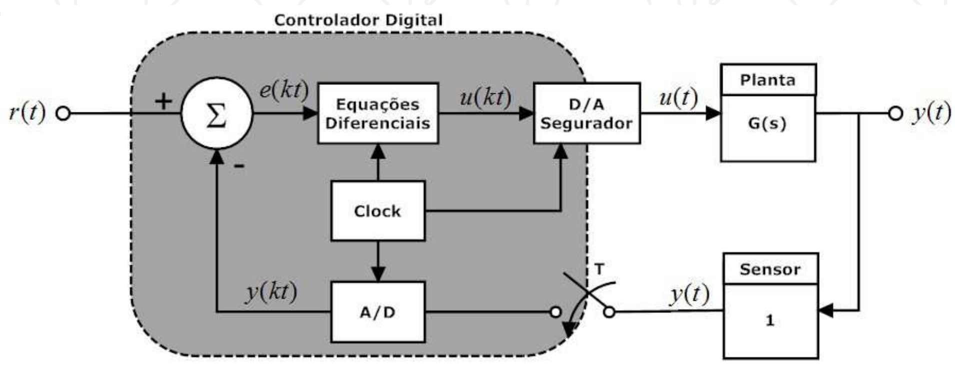


Fig. 1. Digital Control System Diagram

A digital control system is obtained from the reconstruction of analog signal contained in nature. This reconstruction is related mainly with the data acquisition system which is also involved modeling digital control system. Figure 2 illustrates the design of a system basic data acquisition of an industry.

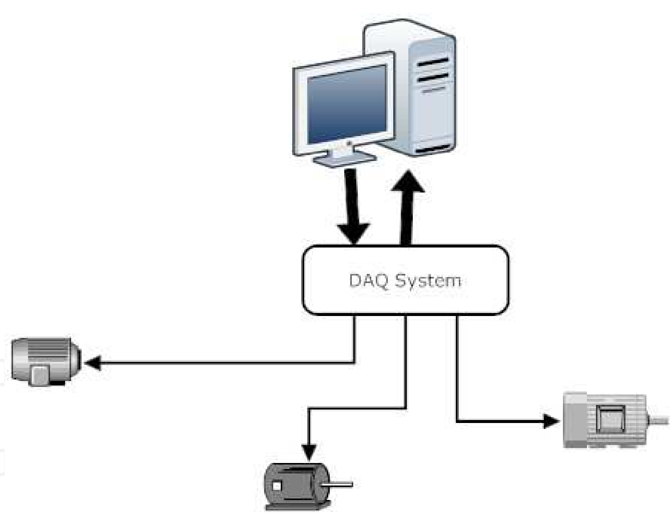


Fig. 2. DAQ System Diagram

The modern control concepts applied in the contemporary industry may be appropriated to the development of academic research activities. The introduction of a methodology parameters that specify and clarify the implementation of monitoring and control of dynamic systems.

In Figures 1 and 2 are shown at different levels, primary architectures to industrial control systems. In Figure 1 is exposed a system with minimal complexity, type SISO (Single Input - Single Output) that can occur in n times or a more complex system can be mapped as a black box system, where internal states of the process do not matter. In Figure 2 is noted,

however, approach of discrete variables on the process that encompasses the entire plant worrying about sub stages.
In this work is explored a way to conceptualize these types of sound architecture in a way such that the system can be modeled and represented whatever its complexity, thus using a powerful tool when it comes to development concepts and analysis.

3. Discrete models of dynamic systems

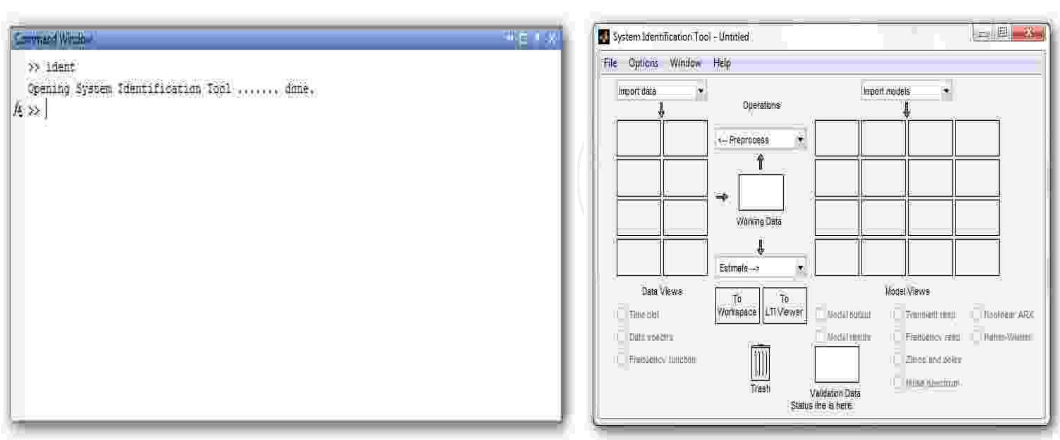
To be held control of dynamic systems, it is necessary in this environment, the perception of some step that must be met for a procedure with minimum guarantees for the project. In the following sections will show important steps as ways to implement them by programming or graphical interface in *MATLAB*, Moudgalya (2007) and Charles L. Philips (1995).

3.1 Dynamic system identification tools

The identification of dynamic systems can be facilitated so as to make procedures the control design more efficient from the standpoint of technical feasibility and cost. processes of various types can be considered as chemical, mechanical or even behavior, used for macroeconomic modeling systems.
The use of tools that comply with the practicality of applying the concept and system design digital control is very suitable as mentioned before, so we introduce a way simple to obtain mathematical models of whatever the process, however in the case of linear or linearized process. To initiate an identification in the *MATLAB* we can proceed in two ways and the designer must choose whichever is most convenient to your time and level of knowledge that it is your process or your plan.
In *Command Window* use the following command to open the graphical user interface the *Identification Toolbox*:

```
>> ident
```

will produce the screens shown in Figure 3, where you can begin the process of identification.

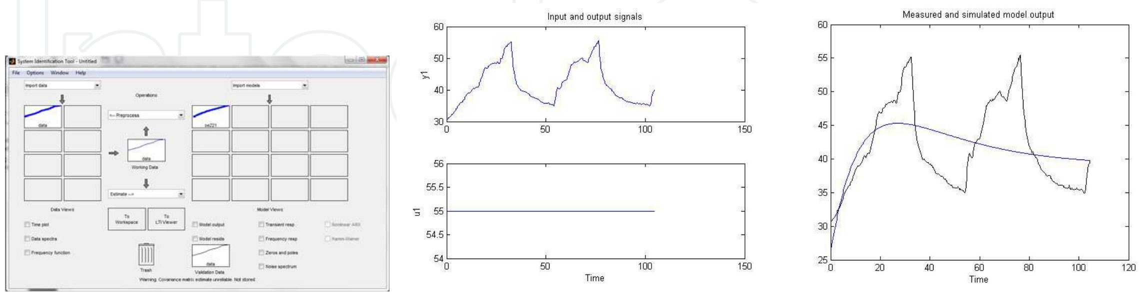


(a) Control to start *Ident* (b) *Ident* Main Screen

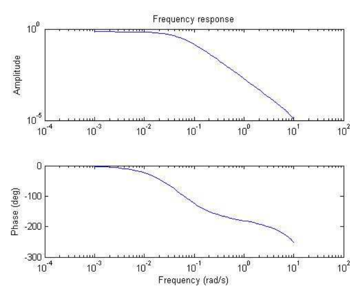
Fig. 3. Opening The Identification Tool Box - *MATLAB*

On these screens you can start importing the measurement data on the system or even with knowledge in the previous system

The main tool for modeling dynamic systems to digital control is the *MATLAB* Identification Tool Box of, a feature that helps the designer to seamlessly and generates discrete polynomial transfer functions and the user without the need of programming the algorithms used for system identification. The *IDENT* from *MATLAB Graphical User Interface* is shown in Figures 4 and 3 and in this figure that can be seen are located to the left set of data and the right answer, ie the model to be obtained.



(a) Ident GUI (b) Data - Time Response (c) Model- Time Response



(d) Model -Frequency Response

Fig. 4. Results From The Identification Tool Box - MATLAB

Still in Figure 4 can observe that the user can define various aspects modeling systems such as filtering of data and type of modeling, as well as the analysis of response in time and frequency of the system model, Gene F. Franklin (1998).
Through this tool we can get two different ID type models in the output. The model based on parameters in the output shows a polynomial equation in discrete form
It should be added that among the numerous tools of identification, we will use polynomial will be the identification, because it gives us a response in discrete time, which makes this model implementable on digital computers. With the option to generate a polynomial model identification as described in Eq.1.

$$G(z) = \frac{Y(z)}{U(z)} = \frac{b_0 + b_1 z^{-1} + b_2 z^{-2} + \dots b_n z^{-n}}{1 + a_1 z^{-1} + a_2 z^{-2} + \dots a_n z^{-n}} \tag{1}$$

3.2 The plant modeling

The estimation of parameters of the transfer function of the plant is performed with the *Toolbox* Identification Systems *MATLAB*. Using the observations of voltage and temperature measurements, table,we can get the parameters of a 2*nd* order transfer function.
In Figure 5 to represent the operation points thermal plant for a first evaluation of the algorithm, shows the temperature behavior from departure to variations in operating points.

The bands system operation are separated for purposes of calculation of the parameters of a model of second order. The band Operation of this plant is between 58^0 and $60\ ^0$ as observed in Figure 5.

3.2.1 Computational and hardware setups

It is understood by *Hardware* setups and computational specifications of the hardware and and design specifications that are the model order and delay. Description of procedure for conducting the experiment are presented in next sections.

3.2.2 OE model

Using *Identification Toolbox* to estimate the parameters of the discrete model **OE** (Output-Error) operated in the delay is given by

$$T(t) = \frac{B(q)}{F(q)}\omega(t) + e(t) \tag{2}$$

where

$$A(q) = 1 - 1.52q^-1 + 0.5336q^{-2}$$
$$B(q) = 0.7426q^{-1} - 0.7187q^{-2}$$

Considering the transfer function of the deterministic signal from the ω input we have

$$\frac{T(t)}{\omega(t)} = \frac{B(q)}{F(q)} \tag{3}$$

3.2.3 The transfer function and polynomial form

The transfer function in Z plane is given by

$$\frac{W(z)}{V_a(z)} = \frac{B(z)}{F(z)} = \frac{0.7426z - 0.718}{z^2 - 1.52z + 0.5336} \tag{4}$$

The poles of the transfer function in Z, $pz_1 = 0.9697$ and $pz_1 = 0.5503$is the positive axis that divides the first and fourth quadrants. The zero is given by $z = 0.9678$ and is on top from one pole to the nearest two decimal places.
Applying the transformation *C2D* command of *Toolbox* Control gives the model transfer function continuously in the model time we have

$$\frac{T(s)}{V_a(s)} = \frac{B(s)}{F(s)} = \frac{0.9852s - 0.03224}{s^2 + 0.628s + 0.01837} \tag{5}$$

The poles of the transfer function at s,$pc_1 = -0.5973$and $pc_2 = -0.0308$. The zero at $zc = -0.0327$ nearly cancels the effect pole pc_2.
The *MATLAB scrip execution* has to read the temperature signals and actuator to estimate the parameters a mathematical model that represents the operating point system's thermal system.

```
Sys = OE(Data,[nb nf nk]
```

where
Data is the object containing the information input and output system;
nb is the order of the polynomial $B(q)$ from Equation 1;
nf is the order of the polynomial $F(q)$ from Equation 1;
nk is the input delay

We can also order the knowledge to perform analysis in continuous time by following the identification commands

```
M = idproc(type,value)
```

Thereby generating a frequency domain model type

$$G(s) = Kp \frac{1 + T_z s}{(1 + T_{p1} s)(1 + T_{p2} s)(1 + T_{p3} s)} e^{-T_d s} \tag{6}$$

Where the parameters of the function ***idproc*** are inherent in the system type, order and constant delay.
The graphs in Figure 5 are constructed in accordance with the instructions from design requirements . The first statement stores the vector *medt* values of temperature around the heat source (resistor) and the actuator speed ω. The third instruction is didactic and for implementation is not required to be codified.

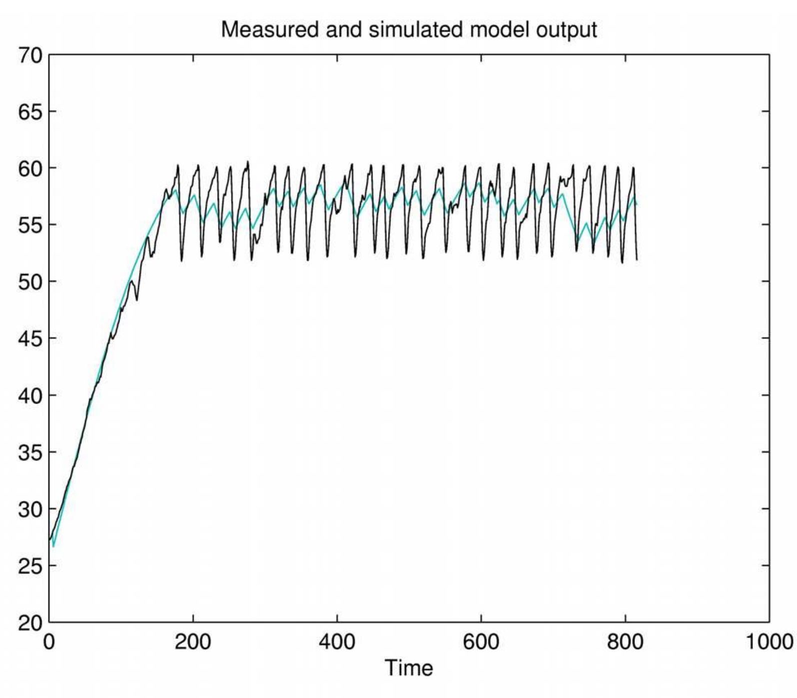


Fig. 5. Continuous Process Reaction Curve.

3.3 Model survey

The lifting of the method consists of parametric estimation conversion this model of transfer function in Z.

3.3.1 Model discrete time

The procedure for the identification of parameters of discrete time is encoded in the *script*, converting (describing) the discrete models in transfer function descriptions.
The Figure 5 shows the comparison of behavior temperature of the continuous model obtained by the process reaction curve

3.3.2 Transfer functions

The purpose of the transfer functions we generate the files contains models of the objects should be stored *OE* to conversion models.

3.3.3 Estimating parameters scripts

The establishment of a procedure for estimating parameters (connotation: scientific) or lifting of a model (connotation: engineering) presented in this section consists of three steps. The first step is the generation each measurement. The second step consists of estimating the parameters of functions transfer. The third step is the analysis of the model have the impulse response, Bode diagram and step.

3.4 A Platform to testing control systems

To validate our experiment we used a platform for experiments with micro digital circuits processed with the support of micro controllers that can be easily programmed using knowledge of language with *C* and a broad support to this type of application as in Ibrahim (2006),Lewis (2001) and Wilmshurst (2007).
Using the Microchip PIC micro controller family we can turn our platform to run on following code done in *C* programming language.

```
#device adc=10
#use delay(clock=4000000)
#fuses HS,NOWDT,PUT
#use rs232(baud=9600, xmit=PIN_C6,rcv=PIN_C7)
#include <mod_lcd.c>

main()
{
   long int value=0;
   float temp=0;
   int i=0;
   lcd_ini();
   setup_timer_2 (T2_DIV_BY_16, 61, 1);
   setup_ADC_ports (RA0_analog);
   setup_adc(ADC_CLOCK_INTERNAL );
   setup_ccp1(ccp_pwm);
   set_pwm1_duty ( 0 );
   setup_ccp2(ccp_pwm);
   set_pwm2_duty ( 0 );
   set_adc_channel(0);
   printf ("%%------------ DAQ - System ------------ \r\n");
   printf ("%%Temperature (žC) \r\n");
   while(true){
```

```
set_pwm1_duty(1023);
 delay_ms(100);
 value = read_adc();
    temp = ((5*(float) value)/(1024))*16.09;
    printf (" %f \r\n",temp);
    printf (lcd_write,"Temp = %f ",temp);
    lcd_write ('\f');

      if(temp>=60){
          set_pwm2_duty ( 1023 );
          }
      if(temp<=58){
      set_pwm2_duty ( 0 );
          }
delay_ms(500);

 }
}
```

The code shown above is the implementation of a system that simulates the temperature, in the form of *Hardware* a thermal system with relative temperature variation in responses to time much smaller.
The dynamics of this system is well demonstrated in Figure 6

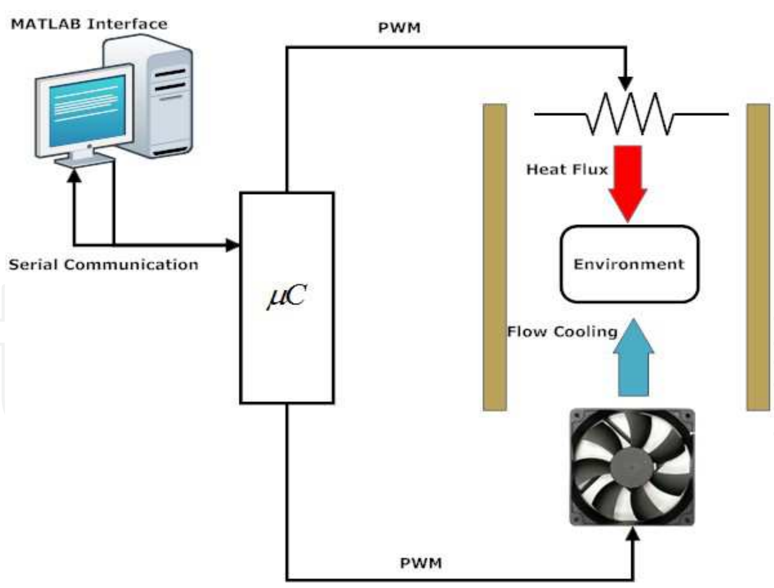


Fig. 6. A Diagram to System Hardware

Where the basis for the system of change of temperature in an open environment that is cooler acts as the system actuator. The control objective, as shown in the code is to keep a room temperature at a desired track with minimal accuracy in the output signal.

4. Implementing control systems on MATLAB

The *MATLAB* software provides support, form the implementation of acquisition system data to design a control system, which becomes possible with methods of identification and communication required in numerical methods implementation of a controller. This tool was incorporated for the purpose to help engineers and scientists in projects and systems design methods numeric, or for the resolution algorithms and systems simulation. In the following sections will be shown tools that are part of the suite of applications such as *MATLAB* concepts for implementation of digital control systems.

4.1 The serial interface

For the design of systems for data acquisition in *MATLAB*, an important tool are the protocols for data communications in environments that are common industries. The serial communication is known for its simplicity of connection and implementation in both hardware and software because there are a wide variety software and programming languages that provide libraries for development communication in serial protocol. In *MATLAB* the simplicity of applying this tool can be translated by code shown below

```
Serial_Obj = Serial('COM1');
fopen(Serial_Obj);
var_read = fscanf(Serial_Obj)
fprintf(var_read,'Data Receiveve');
fclose(Serial_Obj);
Delete(Serial_Obj);
```

With the above script you can communicate and receive data from an external device to your computer. With communication with the *external hardware* can open, using the following commands, send and receive data relevant to the control system

```
function send_data_tohard(data_ctrl,setpoint_ctrl)
fprintf(Serial_Obj,data_ctrl));
fprintf(Serial_Obj,setpoint_ctrl));
```

Therefore the system receives the data for the implementation of intelligent digital control system:

```
function receiv_data_fromhard(data_ctrl,data_error)
data_ctrl = fscanf(Serial_Obj,'data_ctrl');
data_error = fscanf(Serial_Obj,'data_error');
```

Always observing that the variable *Dataa_crtl* is used for timing and appropriateness of real-time system.

4.2 The user interface

The user interface development system shown in Figure 7 is used with supervisors and control environment of the platform.

Using the anointing of the environment development of ***GUIDE*** MATLAB can therefore obtain the convenience of the project environment be the same as the deployment environment

In the source code below is observed in connection with the generation via the serial port textbf GUID emph MATLAB.

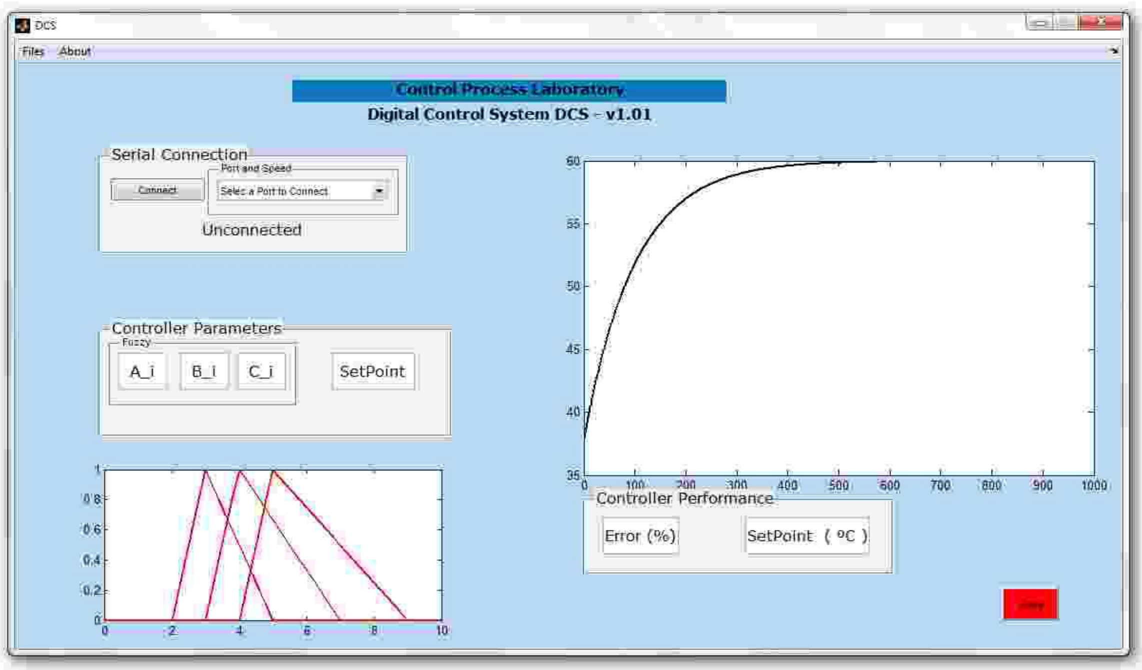


Fig. 7. Graphical Interface of the Platform System

```
function  Button_01_Callback(hObject, eventdata, handles, varargin)
Serial_Obj=serial('COM1','BaudRate',9600);
opcon2 =get(handles.menu_01,'Value');
opcon = get(handles.menu_01,'String');

switch opcon{opcon2}
    case 'COM1'
        Serial_Obj.Port='COM1';
    case 'COM2'
        Serial_Obj.Port='COM2';
    case 'COM3'
        Serial_Obj.Port='COM3';
    case 'COM4'
       Serial_Obj.Port='COM4';
    case 'COM5'
      Serial_Obj.Port='COM5';
    case 'COM6'
       Serial_Obj.Port='COM6';
    case 'COM7'
      Serial_Obj.Port='COM7';
    otherwise
        errordlg('Select a Valid Serial Port','Error')

end

fopen(Serial_Obj);
if Serial_Obj.Status ==open
```

```
set(handles.con_text,'String','Connected');
end
```

4.3 Real time control

Digital Control Systems has its roots in the interface with the analog world and thus the delay time this conversion to occur immediately. In real-time systems is a concern that the execution time of a given instruction does not exceed a predetermined threshold. Converging, the realtime systems are strictly necessary in digital control because it does necessary to guarantee instruction execution control U_c within the limit of sampling system T_s where a failure of this requirement may lead to instability as the plant design of discrete controllers is directly related T_s variable.

4.4 Implementing intelligent systems

The implementation of intelligent systems through *MATLAB* may well become a very profitable it facilitates the testing of new techniques that use features and tools already implemented in this software. Techniques computational intelligence can withdraw from the digital control system the important factor in the T_s, however the system still requires real time control. Each execution cycle lets you have the need to perform numerical derivatives and integrals and introducing a context of researching and mapping , which may require less computational effort. The diagram shown in Figure 8

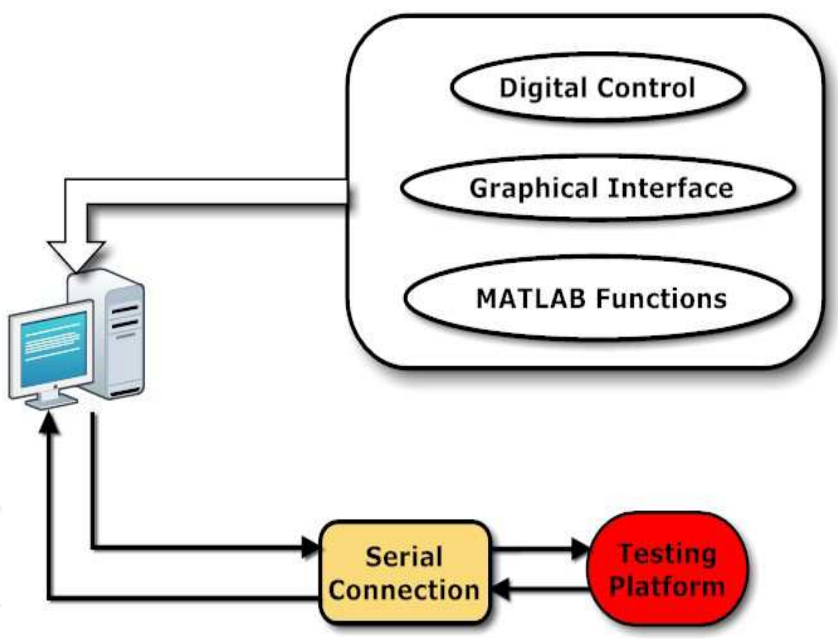


Fig. 8. Diagram of the General System

Itś possible conclude that with the development of both the hardware and software, these systems are likely to become the largest presence within the control and automation

4.4.1 A fuzzy method implementation

As described in Ross (2004),S. Sumathi (2010) and Andrade (2010) the commitment of systems based on fuzzy logic both in respect of the facility as implementation of policies that take into account not only the performance of closed loop of industrial process control as well as the experience acquired human with the process that can be transferred directly to the core of the *Fuzzy Logic Controller*(FLC) with its implementation process described in Figure 9

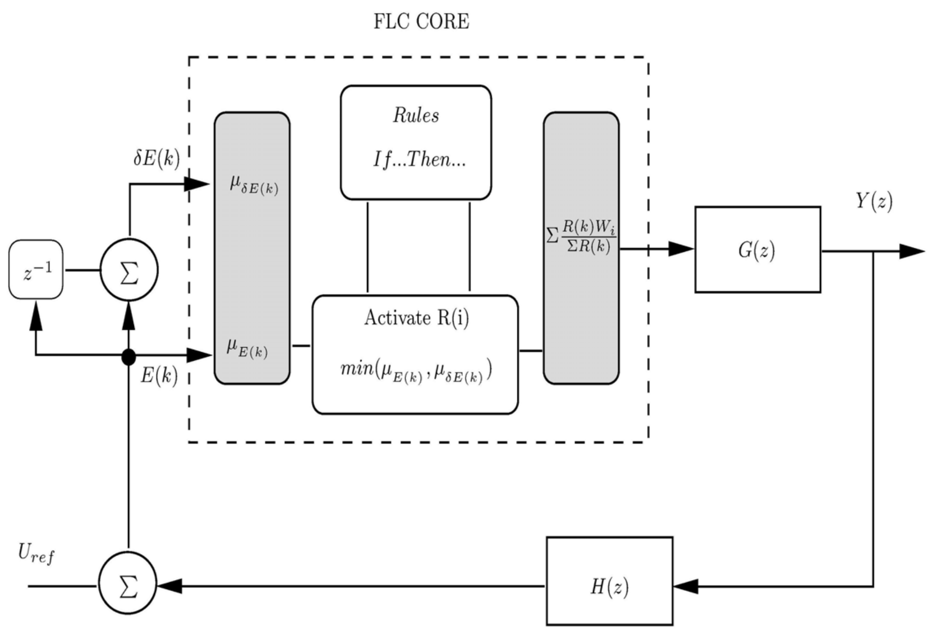


Fig. 9. Diagram of the General System

The consolidation of this type of system could provide designers with a feature not found on controllers classical, ie the aforementioned expertise. To use the Fuzzy controller is needed for the project prior knowledge on the implementation of systems based on fuzzy logic and can be found in Andrade (2010) and especially in Zadeh (1965). In $MATLAB$ the design begins by checking the feasibility of the controller and can be done by graphical interface *Fuzzy Logic Toolbox* with a significant help from MathWorks (2010).
To open the *FLT (Fuzzy Logic Toolbox)* typed:

```
>>fuzzy
```

and the following screen will open
However the system to be implemented with $MATLAB$ using the serial communication is necessary to implement the intelligent system in *script* and can be done as follows

```
error = data_ctrl;
NG(k) = trimf(error, [a_i b_i c_i]);
NS(k) = trimf(error, [a_i b_i c_i]);
ZR(k) = trimf(error, [a_i b_i c_i]);
PS(k) = trimf(error, [a_i b_i c_i]);
PB(k) = trimf(error, [a_i b_i c_i]);
chang_error = data_ctrl;
NL(k) = trimf(chang_error, [a_i b_i c_i]);
NS(k) = trimf(chang_error, [a_i b_i c_i]);
ZR(k) = trimf(chang_error, [a_i b_i c_i]);
PS(k) = trimf(chang_error, [a_i b_i c_i]);
PL(k) = trimf(chang_error, [a_i b_i c_i]);
```

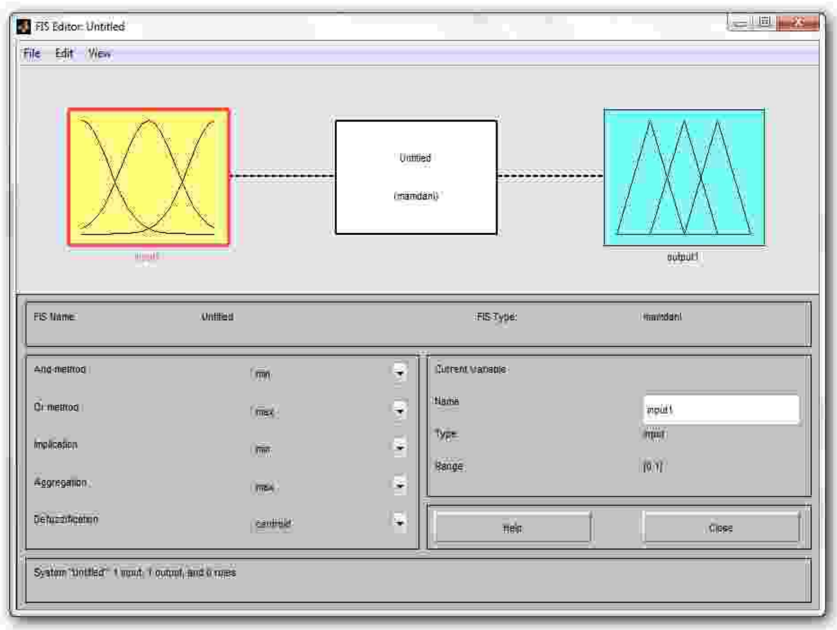


Fig. 10. The FLT Main Screen

Where N*egative* ***L**arge*, ***N**egative* ***S**mall*, ***Z**e**R**o*, ***P**ositive* ***S**mall* and ***P**ositive* *Large* are linguistics vary based on experience. The system output can be given as shown in Figure 9 with a bank of rules based on Table 1 decision-making as an example

Error \ $changing_{E}rror$	**NL**	**NS**	**ZR**	**PS**	**PL**
NL	PL	PL	PL	PL	NL
NS	PS	PS	PS	PS	NS
ZR	PS	ZR	ZR	NS	ZR
PS	NS	NS	NS	NS	PS
PL	NL	NL	NL	NS	PS

Table 1. The knowledge from the Rules of Decision System

And the controller output and results in a weighted sum that depends exclusively on the method chosen.

5. Conclusion

With the aid of this important tool can facilitate the development of applications to interface with and control devices industries. A overview of these tools has been shown here to that may be developed to greater diversity of applications according to the need of the designer. As an environment of the computational complex, *MATLAB* can provide solutions for engineering, physics and mathematics with the use of its functions basic and therefore leaving the search more efficient and less costly. This platform is an important and necessary tool, it can lead a project, with integration between *Software* and *Hardware*, for a safe convergence of results, thus integrating computational resources for simulation design and implementation from a direct single platform.

6. References

Andrade, G.A. Neto, J. L. L. (2010). A framework for modeling, digital control design and simulations of dynamic, *Computer Modelling and Simulation (UKSim), 2010 12th International Conference on*, IEEE Computer Society Digital Library.

Charles L. Philips, H. T. N. (1995). *Digital Control Systems: Analisys and Design*, third edition edn, Prentice Hall.

Gene F. Franklin, J. David Powell, M. L. W. (1998). *Digital Control of Dynamic Systems*, 3rd edn, Prentice Hall.

Ibrahim, D. (2006). *Microcontroller Based Applied Digital Control*, John Wiley and Sons.

Lewis, C. S. (2001). *Programming Microcontrollers in C*, 2nd edn, LLH Publishing.

MathWorks, T. (2010). *Fuzzy Logic Tool Box : User's Guide*.

Moudgalya, K. (2007). *Digital Control*, John Wiley and Sons.

Ross, T. J. (2004). *Fuzzy Logic With Engineering Applications*, 2nd edn, John Wiley and Sons.

S. Sumathi, S. P. (2010). *Computational Intelligence Paradigms: Theory and Applications Using MATLAB*, CRC Press.

Wilmshurst, T. (2007). *Designing Embedded Systems with PIC Microcontrollers*, Newnes.

Zadeh, L. A. (1965). Fuzzy sets, *Information and Control Proceedings* .

MATLAB - A Ubiquitous Tool for the Practical Engineer
Edited by Prof. Clara Ionescu

ISBN 978-953-307-907-3
Hard cover, 564 pages
Publisher InTech
Published online 13, October, 2011
Published in print edition October, 2011

A well-known statement says that the PID controller is the â€œbread and butterâ€ of the control engineer. This is indeed true, from a scientific standpoint. However, nowadays, in the era of computer science, when the paper and pencil have been replaced by the keyboard and the display of computers, one may equally say that MATLAB is the â€œbreadâ€ in the above statement. MATLAB has became a de facto tool for the modern system engineer. This book is written for both engineering students, as well as for practicing engineers. The wide range of applications in which MATLAB is the working framework, shows that it is a powerful, comprehensive and easy-to-use environment for performing technical computations. The book includes various excellent applications in which MATLAB is employed: from pure algebraic computations to data acquisition in real-life experiments, from control strategies to image processing algorithms, from graphical user interface design for educational purposes to Simulink embedded systems.

How to reference
In order to correctly reference this scholarly work, feel free to copy and paste the following:

João Viana da Fonseca Neto and Gustavo Araújo de Andrade (2011). MATLAB as Interface for Intelligent Digital Control of Dynamic Systems, MATLAB - A Ubiquitous Tool for the Practical Engineer, Prof. Clara Ionescu (Ed.), ISBN: 978-953-307-907-3, InTech, Available from: http://www.intechopen.com/books/matlab-a-ubiquitous-tool-for-the-practical-engineer/matlab-as-interface-for-intelligent-digital-control-of-dynamic-systems