DISEÑO INTERACTIVO DE INTERFACES DE USUARIO PARA SISTEMAS DE CONTROL USANDO MATLAB

Bruno Vargas Tamani
bvargast@unmsm.edu.pe

Facultad de Ingeniería Electrónica
Universidad Nacional Mayor de San Marcos

RESUMEN: El presente artículo describe el uso y utilidad del módulo Entorno de Desarrollo de Interfaz Gráfico de Usuario (Graphical User Interface Development Environmet-GUIDE) de MATLAB (v. 5.3). Se explica los objetos con los que se cuenta, sus propiedades más importantes, sus bloques, la manera de acceder a ellos y modificar sus propiedades a fin de lograr el diseño deseado. Se presentan ejemplos de aplicación en ingeniería de control, que servirán como base para otras aplicaciones similares.

ABSTRACT: This paper describe the use and utility of the module of MATLAB (v. 5.3), which has been called Graphical User Interface Development Environmet (GUIDE). It is explained the objects, the more important properties, the blocks, the way to enter to them and to modify the properties in order to achieve the design desired. Examples of application in engineering of control are presented, that will serve like base for other similar applications.

I. INTRODUCCIÓN

Al usar el programa de cálculo numérico basado en vectores y matrices de MATLAB para realizar análisis y diseños de diferentes tipos de problemas en ingeniería de control para lo cual se debe ejecutar más de un programa a nivel de comandos agrupados en un archivo tipo **.m* y simular los resultados previos y finales obtenidos, usando el toolbox SIMULINK del mismo MATLAB, se generan diferentes archivos que constituyen un todo para la solución de un problema. A partir de estos archivos se puede crear una interfaz gráfica de usuario que presenta organizadamente los resultados de análisis y diseño de cada etapa, así como botones, menús, ventanas, etc., que permiten la ejecución de los programas del proyecto en la secuencia elegida, así como presentar resultados en forma gráfica, etc. Para este fin, MATLAB incorpora un módulo denominado GUIDE (Graphical User Interface Development Environmet), que permite crear de modo interactivo, la Interfaz de Usuario.

II. ESTRUCTURA DE GRAFICOS EN MATLAB

Los gráficos en MATLAB están compuestos de objetos, siendo el más general la pantalla (*screen*) [García, 1999]. La pantalla puede contener una o más ventanas (*figures*), cada una puede contener uno o más ejes de coordenadas (*axes*), así como controles (*uicontrol*) como: botones (*pushbuttons*), barras de desplazamiento (*scrolling* bar o *sliders*), botones de selección (*check boxes*), botones de opción (*radio buttons*), cajas de texto (*static textboxes*), cajas de texto editables (*editable textboxes*), marcos (*frames*), cajas de selección desplegables (*pop-up menus*) y menús (*menus*). Cada eje puede contener líneas (*lines*), rectángulos (*rectangles*), polígonos (*patches*), superficies (*surfaces*), imágenes (*bitmat images*) y texto (*text*).

Cada objeto tiene un identificador único (*handle*). Siempre hay una sola ventana activa y en ella sólo un eje activo. Los gráficos se presentan en el eje activo de la ventana activa. Para conocer el *handle* de la ventana activa, del eje activo y del objeto activo se deben usar los comandos gcf (*get current figure*), gca (*get current axes*), gco (*get current object*). Todos los objetos tienen distintas propiedades, las cuales tienen valores por de-

fecto, algunas modificables y otras no. Usando las funciones *set* y *get* se pueden consultar y cambiar las propiedades de un objeto.

III. PROPIEDADES DE LOS OBJETOS

A continuación se describe brevemente las propiedades más importantes de los objetos:

- *String*: texto mostrado en el objeto.
- *Label*: propiedad del texto del menú.
- *Tag*: nombre interno del objeto. Se puede usar para ubicar ese objeto.
- *Style*: clase de objeto.
- *Position*: vector de cuatro elementos (*left, botton, width, height*), que a partir del origen (esquina inferior izquierda), indica la posición y tamaño del objeto.
- *Units*: unidades de las dimensiones.
- *BackgroundColor*: vector de tres elementos, que indican las componentes RGB del color de fondo del objeto.
- *ForegroundColor*: vector de tres elementos, que indican las componentes RGB del color de texto del objeto.
- *Enable*: indica si el control está activo o no.
- Visible: indica si el control es visible o no.
- *FontName*: el tipo de letra a utilizar.
- *FontSize*: el tamaño de letra.
- *FontWeigth*: puede ser *normal, ligth, demi* ó *bold.*
- *UserData*: dato de usuario que quiera asociar con el control.
- *Value*: valor asociado con algunos controles. Como posición en la barra de desplazamiento; valor de propiedad max cuando están en *on* y *min* cuando están en *off* en los *checkbuttons* y *radiobuttons*; número ordinal del elemento seleccionado en los *listbox* y *popupmenus.*

IV. CREACIÓN DE INTERFACES DE USUARIO EN FORMA INTERACTIVA

Se crea la Interfaz de Usuario ejecutando el módulo GUIDE desde la línea de comandos, escribiendo el comando *guide*. Seguidamente aparecerá la pantalla *Guide Control Panel*, como la mostrada en la figura 1. Además, aparece una ventana (*figure*) en blanco sobre la cual se coloca los ejes y controles que se requieran usando el mouse.

Hay cuatro íconos ubicados en la parte superior de la pantalla, presionando sobre el ícono Editor de Propiedades (*Property Editor*) se abre una ventana como la mostrada en la figura 2, en donde se muestra las listas de los objetos y la de propiedades del objeto seleccionado, desde aquí se puede modificar las características de los objetos.

El ícono Editor de llamadas (*Callback Editor*), presenta una ventana como la mostrada en la figura 3, se puede programar todo los comandos que se desee ejecutar al realizar alguna acción sobre algún control de la Interfaz de Usuario.

También, se puede usar las funciones definidas por el usuario, las que contienen el código a ejecutarse, ésta es la forma más conveniente de usar el *editor* en la que se muestra la lista de los objetos y la lista de las propiedades del objeto seleccionado, a su vez es posible modificar las características de los objetos.

El editor de alineamiento (*Alignment Editor*), permite alinear y distribuir los controles de una ventana. El Editor de Menús (*Menu Editor*), permite crear menús adicionales a los que presentan todas las ventanas por defecto, que son: *File*, *Edit*, *Windows* y *Help*.

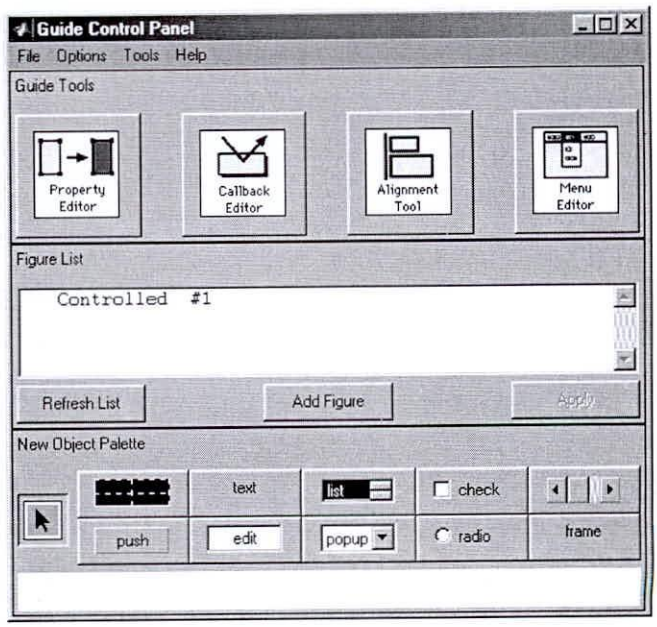

Figura 1 - Pantalla Guide Control Panel (GCP).

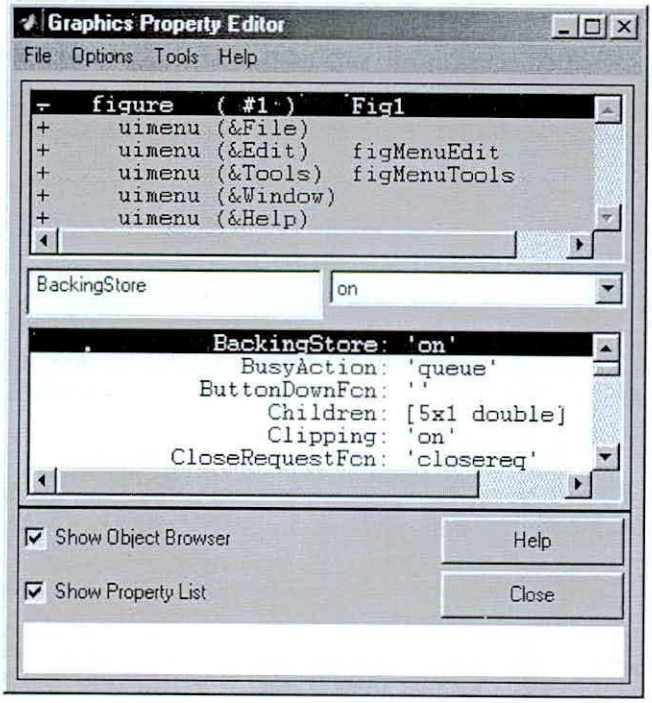

Figura 2 - Editor de Propiedades (Property Editor)

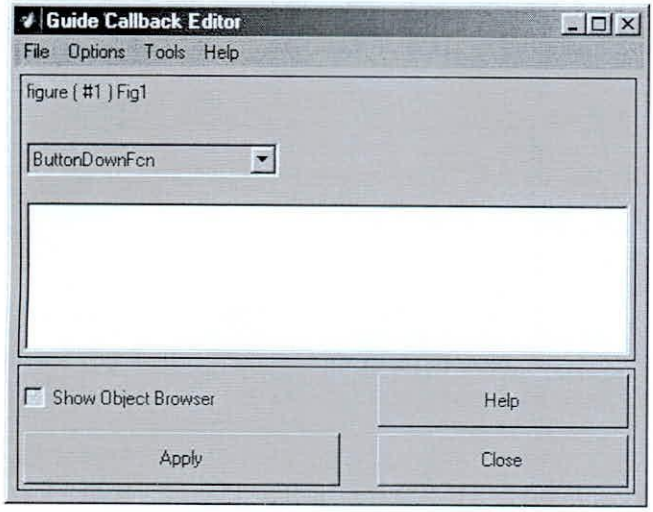

Figura 3 - Editor de llamadas (Callback Editor).

V. EJEMPLO 1 - INTERFAZ DE USUARIO PARA LA SIMULACIÓN DE UN CONTROL DE PROCESOS

Este ejemplo, presenta el diseño de una simple Interfaz de Usuario, que permita realizar la prueba de un sistema de lazo cerrado del control del flujo de una tubería usando como actuador a una válvula del tipo isoporcentual [Creus, 1998].

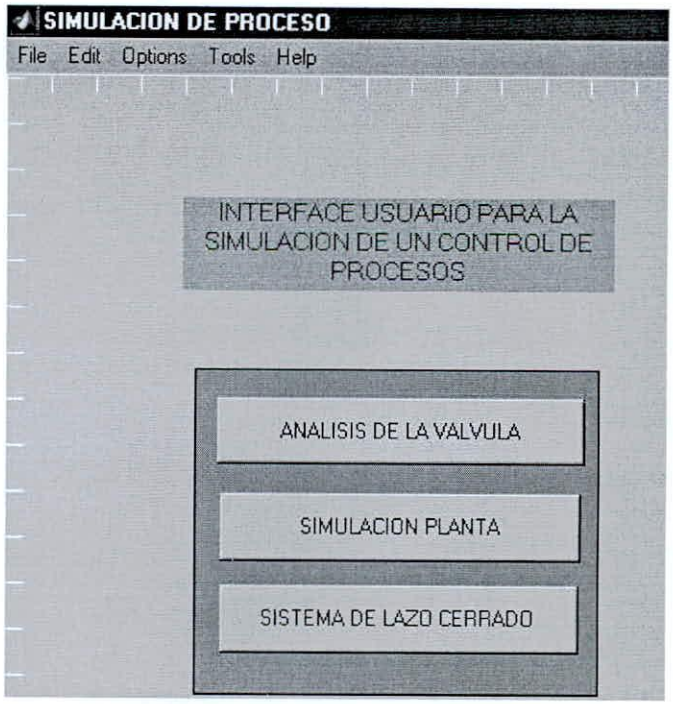

Figura 4 - Interfaz de Usuario para simular el del control de flujo de una tubería.

El proceso de diseño se inicia conociendo el funcionamiento del modelo del sistema de tuberías, el retardo que se presenta, así como el modelo del conversor de corriente a presión que es necesario usar. Se utiliza un controlador PID, cuyo comportamiento se verifica a través del simulador implementado por una Interfaz de Usuario.

Ha sido necesario estudiar la característica estática de la válvula, así como de la planta en lazo abierto y probar el comportamiento en lazo cerrado del PID. Así, para cada situación se implementó en Simulink. el respectivo programa de simulación, los que son invocadas por el usuario.

La Interfaz, ver figura 4, consta de una caja de texto (con un titulo) y tres botones. Al hacer clic en cada botón se presenta, según sea el caso, el diagrama en Simulink del análisis de la válvula, la simulación de la planta o de la simulación del sistema de lazo cerrado.

5.1 Propiedades de los Objetos

- Propiedades modificadas de la ventana
  Name 'SIMULACION DE PROCESO', NumberTitle off.

- Propiedades modificadas de caja de texto BackgroundColor [010], FontSize 12, ForegroundColor [001], String 'INTERFAZ DE USUARIO PARA LA SIMULACION DE UN CONTROL DE PROCESOS'.
- Propiedades del marco son por defecto.
- Propiedades modificadas del primer botón String 'ANALISIS DE LA VALVULA'
- Propiedades modificadas del segundo botón String 'SIMULACION PLANTA'
- Propiedades modificadas del tercer botón String 'SISTEMA DE LAZO CERRADO'

El Editor de llamadas *Callback Editor*, para cada control contiene lo siguiente:

- Para el primer botón: open valvula.mdl
- Para el segundo botón: open proc.mdl
- Para el tercer botón: open procesoflujo.mdl

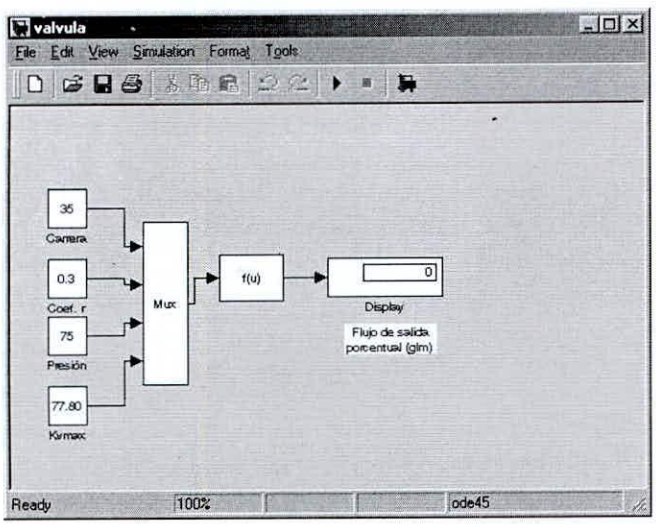

Figura 5 - Diagrama de Simulink para la opción Válvula.

Los archivos *.mdl, corresponden a los diagramas de simulación programados en *Simulink*, los que se abrirán al pulsar con el mouse sobre los botones correspondientes. Las figuras 5, 6 y 7 muestran los diagramas que corresponden a cada una de las opciones que el usuario puede seleccionar en la Interfaz de Usuario.

Esta simple Interfaz de Usuario fácil de diseñar, permitió realizar las simulaciones con sólo hacer clic en el botón apropiado. Luego, dentro del diagrama del Simulink se puede modificar los datos de entrada según convenga.

VI. EJEMPLO 2 - LLENADO DE DOS TANQUES

Se presenta una animación, cuyo efecto da la impresión del llenado de dos tanques comunicados a través de una tubería [Roca, 1999]. El flujo de salida de cada tanque varía en función a las resistencias de las válvulas correspondientes. La magnitud del flujo de alimentación que llega al primer tanque puede ser ajustada modificando la corriente de entrada de la electroválvula.

El usuario puede modificar en cualquier momento la corriente de entrada de la electroválvula, para ello se utilizó un programa basado en el modelo en espacio de estado del sistema, el cual se invoca al cambiar el valor de la barra de desplazamiento que se usará en la Interfaz de Usuario que corresponderá a la nueva corriente de entrada.

Se usa un botón para el inicio del llenado de los tanques, cada vez que se actualice el valor de corriente. La Interfaz de Usuario consta de una ventana con cuatro ejes, en donde se gráfica las imágenes asociadas con las dos válvulas y con los dos tanques.

En el eje de tanques se programó la animación del llenado de ambos. Además, se tiene dos cajas de texto (para colocar el valor de la mínima y máxima corriente de entrada), la barra de desplazamiento (para ingresar el valor de la nueva corriente) y dos botones (para las acciones de inicio del llenado y fin del programa).

El aspecto de la Interfaz de Usuario antes de iniciar el llenado se presenta en la figura 8 y la Interfaz de Usuario luego de la animación del llenado de los tanques se muestra en la figura 9.

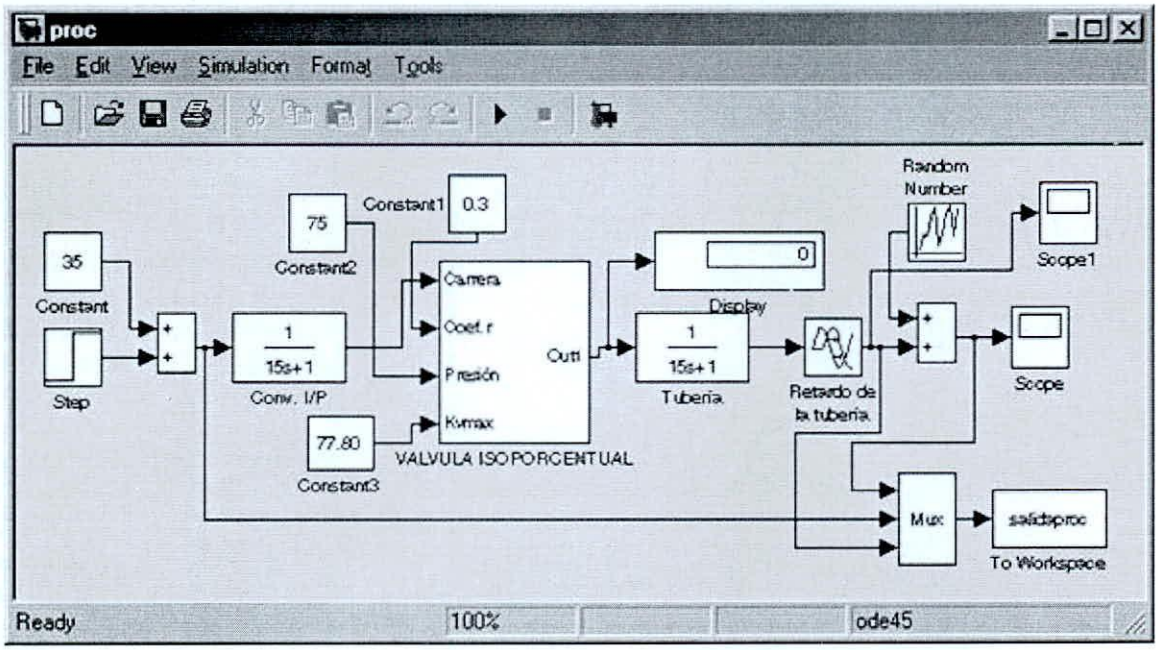

Figura 6 - Diagrama de Simulink para la opción Simulación Planta.

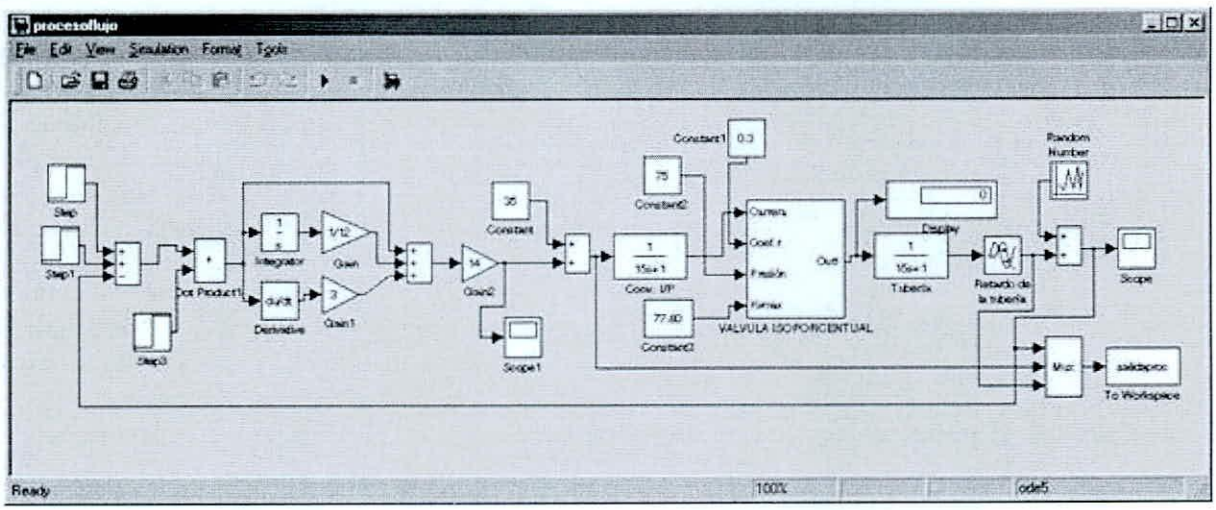

Figura 7 - Diagrama de Simulink para la opción Sistema de Lazo Cerrado.

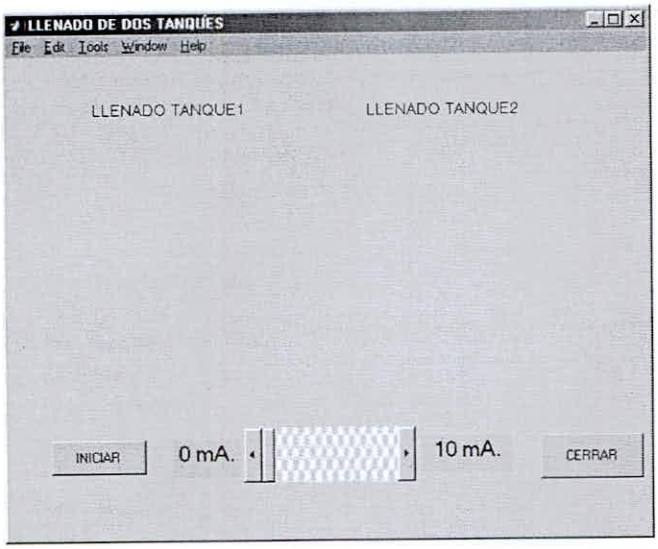

Figura 8 - Aspecto de la Interfaz de Usuario antes de iniciar el llenado.

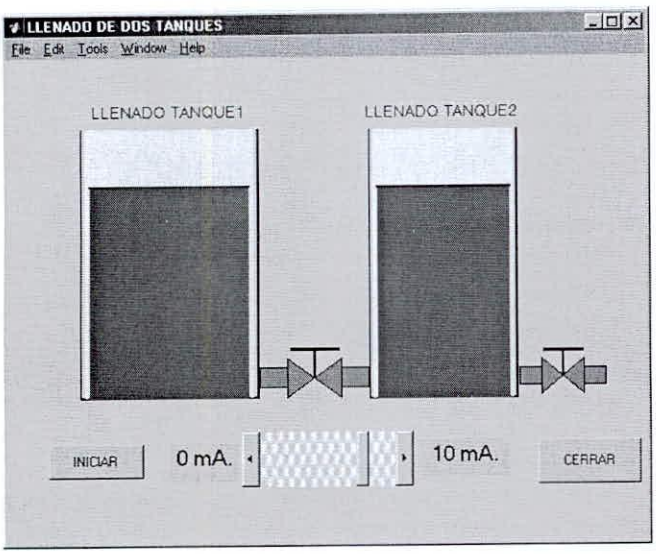

Figura 9 - Interfaz de Usuario luego de la animación del llenado.

6.1 Propiedades de los Objetos

- Propiedades modificadas de la ventana
  Name 'LLENADO DE DOS TANQUES', NumberTitle off, Tag 'Fig1'.
- Propiedades modificadas de caja de texto 1
  FontSize 14, String '0 mA.'.
- Propiedades modificadas de caja de texto 2
  FontSize 14, String '10 mA.'.
- Propiedades modificadas de la barra de desplazamiento
  Min 0, Max 0.010, Tag 'nivel2'.
- Propiedades modificadas del botón 1
  String 'INICIAR'.
- Propiedades modificadas del botón 2
  String 'CERRAR'.
- Propiedades modificadas del eje1
  Tag 'tanque1', Visible off. Dentro de este eje el objeto text (title) debe tener propiedad: String 'LLENADO TANQUE1', Visible on.
- Propiedades modificadas del eje2
  Tag 'tanque2', Visible off. Dentro de este eje el objeto text (title) debe tener propiedad: String 'LLENADO TANQUE2', Visible on.
- Propiedades modificadas del eje3
  Tag 'valv1', Visible off.
- Propiedades modificadas del eje4
  Tag 'valv2', Visible off.

El Editor de llamadas *Callback Editor*, para cada control contiene lo siguiente:

- Para el primer botón (iniciar): *tank1; tank2*
- Para el segundo botón (cerrar): *close*

Los archivos *tank1.m* y *tank2.m*, se ejecutan en esa secuencia al pulsar con el mouse en el botón *iniciar* de la Interfaz de Usuario. Ambos archivos permiten realizar la animación deseada, luego que el usuario modifique el valor de corriente de entrada. En el anexo del presente artículo se muestran la edición de ambos programas. Al pulsar el botón cerrar, se ejecuta el comando *close*, con lo cual se cierra la Interfaz de Usuario.

VII. EJEMPLO 3 - ANÁLISIS DE DIFERENTES TIPOS DE ESTABILIDAD

En este ejemplo, se realiza un análisis de los diferentes tipos de estabilidad que se pueden presentar. Para ello se usó un modelo de espacio de estado de segundo orden, del cual se analiza su comportamiento en el plano de fase al iniciar su dinámica con diferentes valores de vectores iniciales (Hassan, 1996). Además, se muestra la variación temporal de los estados para las mismas condiciones iniciales.

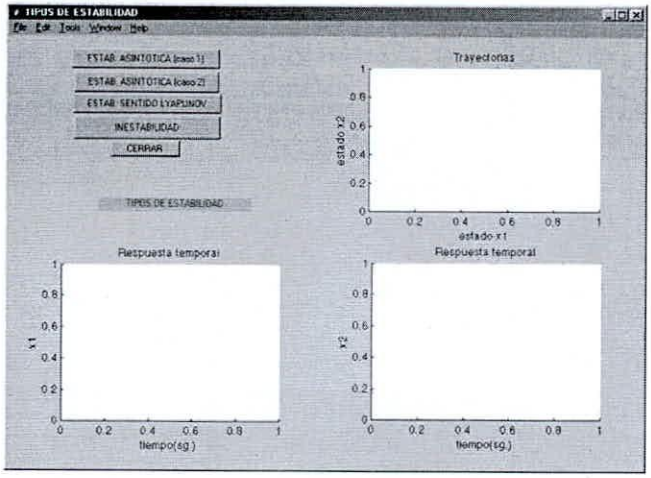

Figura 10 - Interfaz de Usuario inicial (a la espera de escogerse un tipo de estabilidad).

La figura 10, muestra la Interfaz de Usuario, permite elegir al usuario, el tipo de estabilidad que desea analizar y presentar las trayectorias en el plano de fase del sistema, así como la variación temporal de las variables de fase, mostrando además los polos del sistema para cada tipo de estabilidad. La figura 11, presenta la Interfaz de Usuario, que presenta el resultado del análisis del tipo de estabilidad elegido.

Se requirió adicionar a la Interfaz de Usuario dos cajas de texto (una que indique el tipo de estabilidad escogida y otra que muestre el valor de los polos del sistema en esa situación), cinco botones (cuatro para seleccionar el tipo de estabilidad y el último para cerrar el programa), tres ejes (uno para mostrar la variación de las trayectorias, otro para mostrar las variaciones temporales del primer estado y la tercera para mostrar las variaciones temporales del segundo estado).

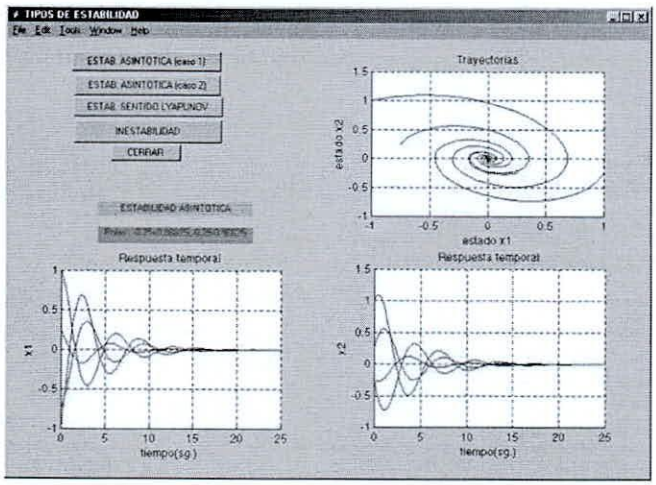

Figura 11 - Interfaz de Usuario, presenta resultados del análisis elegido.

7.1 Propiedades de los Objetos

- Propiedades modificadas de la ventana
  Name 'TIPOS DE ESTABILIDAD', NumberTitle off, Tag 'Fig1'.
- Propiedades modificadas de caja de texto 1 BackgroundColor [010], Visible off, Tag 'polos'.
- Propiedades modificadas de caja de texto 2 BackgroundColor [011], Visible on, Tag 'tipo'.
- Propiedades modificadas del botón 1
  String 'ESTAB. ASINTOTICA (caso 1)'.
- Propiedades modificadas del botón 2
  String 'ESTAB. ASINTOTICA (caso 2)'.
- Propiedades modificadas del botón 3
  String 'ESTAB. SENTIDO LYAPUNOV'.
- Propiedades modificadas del botón 4
  String 'INESTABILIDAD'.
- Propiedades modificadas del botón 5
  String 'CERRAR'.
- Propiedades modificadas del eje1
  Tag 'ejes1', Visible on. Dentro de este eje el objeto text title debe tener propiedad: String 'Trayectorias'; el objeto text xlabel debe tener propiedad: String 'estado x1'; el objeto text ylabel debe tener propiedad: String 'estado x2'.
- Propiedades modificadas del eje2
  Tag 'ejes2', Visible on. Dentro de este eje el objeto text title debe tener propiedad: String 'Respuesta temporal'; el objeto text xlabel debe tener propiedad: String 'tiempo(sg.)'; el objeto text ylabel debe tener propiedad: String 'x1'.
- Propiedades modificadas del eje3
  Tag 'ejes2', Visible on. Dentro de este eje el objeto text title debe tener propiedad: String 'Respuesta temporal'; el objeto text xlabel debe tener propiedad: Strin'tiempo(sg.)'; el objeto text ylabel debe tener propiedad: String 'x2'.

El Editor de llamadas *Callback Editor*, para cada control contiene lo siguiente:

- Para el primer botón

('ESTAB. ASINTOTICA (caso 1) '): A=[-1 0;1 -2];B=[0; 0];C=eye(2);D=zeros(2,1);mod=ss(A,B,C,D);estinic=[-1 -1 1 1 0.5 -0.5 0.25 -0.25 1 -1;1 0.25 -1 -0.25 -0.5 0.5 -0.75 0.75 1 -1]; polo=eig(A);graf(mod,estinic,'ESTABILIDAD ASINTOTICA',polo);

- Para el segundo botón ('ESTAB. ASINTOTICA (caso 2)'): A=[0 1;-1 -0.5];B=[0; 0];C=eye(2);D=zeros(2,1);estinic=[-1 -0.75 1 0.25;1 0.25 -0.25 -0.25];mod=ss(A,B,C,D); polo=eig(A);graf(mod,estinic,'ESTABILIDAD ASINTOTICA', polo);

- Para el tercer botón ('ESTAB. SENTIDO LYAPUNOV'): A=[0 1;-1 0];B=[0; 0];C=eye(2);D=zeros(2,1);estinic=[-1 -0.75 1 0.25;1 0.25 -0.25 -0.25];mod=ss(A,B,C,D); polo=eig(A);graf(mod,estinic,'ESTAB. SENTIDO LYAPUNOV', polo);

- Para el cuarto botón ('INESTABILIDAD'): A=[0 1;2 -1];B=[0; 0];C=eye(2);D=zeros(2,1); estinic=[-1 -1 1 1 0.5 -0.5 0.25 -0.25 1 -1;1 0.25 -1 -0.25 -0.5 0.5 -0.75 0.75 1 -1]; mod=ss(A,B,C,D); polo=eig(A);graf(mod,estinic,'INESTABILIDAD', polo);

- Para el quinto botón (cerrar): close.

En todos los casos la ejecución de comandos, activa una función llamada graf.m, cuya lista de comandos se edita en los anexos.

VIII. CONCLUSIONES

Es muy sencillo realizar una Interfaz de Usuario en MATLAB, se recomienda que cuando realice un análisis, un diseño o un proyecto, organice sus archivos o componentes para que posteriormente puedan ser utilizados adecuadamente a través de una Interfaz de Usuario. El uso del módulo GUIDE, permite crear interactivamente interfaces, ofreciéndonos evitar la dificultad de un diseño a través de comandos más complicados para manejar las propiedades de los objetos, opción única que existía en el pasado.

IX. ANEXOS

A.1.- Edición de archivo tank1.m

```
%tank1.m
function graf(modelo,xin,tipoest,eigen)
h=findobj(0,'tag','tank2')%
h=findobj(0,'tag','tanque1');axes(h);
xtank=[0 0 1 1];ytank=[1 0 0 1];
line(xtank,ytank,'linewidth',2,'color','black')
h=findobj(0,'tag','tanque2');axes(h)
line(xtank,ytank,'linewidth',2,'color','black')
h=findobj(0,'tag','valv1');axes(h)
set(h,'visible','off')
patch([0 0 0.25 0.25 0],[0.3 0.1 0.1 0.3 0.3],[0.5 0.5 0.5])
patch([0.75 0.75 1 1 0.75],[0.3 0.1 0.1 0.3 0.3],[0.5 0.5 0.5])
patch([0.25 0.25 0.75 0.75 0.25],[0.4 0 0.4 0 0.4],[0.5 0.5 0.5])
line([0.5 0.5],[0.2 0.5],'linewidth',2,'color','black')
line([0.3 0.7],[0.5 0.5],'linewidth',4,'color','black')
h=findobj(0,'tag','valv2');axes(h)
set(h,'visible','off')
patch([0 0 0.25 0.25 0],[0.3 0.1 0.1 0.3 0.3],[0.5 0.5 0.5])
patch([0.75 0.75 1 1 0.75],[0.3 0.1 0.1 0.3 0.3],[0.5 0.5 0.5])
patch([0.25 0.25 0.75 0.75 0.25],[0.4 0 0.4 0 0.4],[0.5 0.5 0.5])
line([0.5 0.5],[0.2 0.5],'linewidth',2,'color','black')
line([0.3 0.7],[0.5 0.5],'linewidth',4,'color','black')
h=findobj(0,'tag','tanque2');axes(h)
patch([0.025 0.025 0.975 0.975 0.025],[1 0.0 0.0 1 1],[0.9 0.9 0.9],'edgecolor','none')
h=findobj(0,'tag','tanque1');axes(h)
patch([0.025 0.025 0.975 0.975 0.025],[1 0.0 0.0 1 1],[0.9 0.9 0.9],'edgecolor','none')
```

A.2.- Edición de archivo tank2.m

```
%tank2.m
A1=2;R1=150;A2=1;R2=100;
A=[-1/(A1*R1) 1/(A1*R1);1/(A2*R1) -(1/A2)*(1/R1+1/R2)];B=[1/A1;0];
xin=[0;0];x=xin;xant=xin;uant=0;dt=1;i=1;nivel2ant=0;
nivel1ant=0;
while(1)
h=findobj(0,'tag','nivel2');
u=get(h,'value');
```

```
nivel2=(B(1)/(A(1,1)*A(2,2)/A(2,1)-A(1,2)))*u;
nivel1=(-A(2,2)/A(2,1))*nivel2;
 if(u~=uant)
   h=findobj(0,'tag','tanque1');axes(h)
   if (nivel1<x(1))
     line([.05 .95],([nivel1ant
nivel1ant])/2.5,'linewidth',2,'color',[0.9 0.9 0.9]);
   else
     line([.05 .95],([nivel1ant
nivel1ant])/2.5,'linewidth',2,'color','red');
     end
     uant=u;
  h=findobj(0,'tag','tanque2');axes(h)
  if (nivel2<x(2))
    line([.05 .95],[nivel2ant
nivel2ant],'linewidth',2,'color',[0.9 0.9 0.9])
  else
    line([.05 .95],[nivel2ant
nivel2ant],'linewidth',2,'color','red')
  end
  end
sal1(i)=x(1);sal2(i)=x(2);
xpto=A*x+B*u;
x=x+dt*xpto;
  if (20*floor(i*dt/20)==i*dt)
  h=findobj(0,'tag','tanque1');axes(h)
  if xpto(1)>=0
patch([0.05 0.05 0.95 0.95 0.05],([x(1) xant(1) xant(1)
x(1) x(1)])/2.5,'red','edgecolor','none')
    else
      patch([0.05 0.05 0.95 0.95 0.05],([x(1) xant(1)
xant(1) x(1) x(1)])/2.5,[0.9 0.9 0.9],'edgecolor','none')
    end
    line([.05 .95],([nivel1
nivel1])/2.5,'linewidth',2,'color','blue')
 pause(0.05)
h=findobj(0,'tag','tanque2');axes(h)
   if xpto(2)>=0
    patch([0.05 0.05 0.95 0.95 0.05],[x(2) xant(2)
xant(2) x(2) x(2)],'red','edgecolor','none')
    else
      patch([0.05 0.05 0.95 0.95 0.05],[x(2) xant(2)
xant(2) x(2) x(2)],[0.9 0.9 0.9],'edgecolor','none')
    end
    line([.05 .95],[nivel2
nivel2],'linewidth',2,'color','blue')
    xant=x;
 end
i=i+1;
nivel2ant=nivel2;
nivel1ant=nivel1;
end
```

A.3.- Edición de archivo graf.m

```
%graf.m
function graf(modelo,xin,tipoest,eigen)
clear y
clear t
h=findobj(0,'tag','tipo');
set(h,'string',tipoest);
h=findobj(0,'tag','polos');
set(h,'visible','on');
mostrar=['Polos : ' num2str(eigen(1)) ','
num2str(eigen(2))];
set(h,'string',mostrar);
axes(findobj(0,'tag','ejes3'));
cla
axes(findobj(0,'tag','ejes2'));
cla
axes(findobj(0,'tag','ejes1'));
cla
h=findobj(0,'tag','tipo');
hold on
for i=1:length(xin)
  y=initial(modelo,xin(:,i));
  plot(y(:,1),y(:,2));
  clear y
  pause(1)
end
grid on
disp('Pulse una tecla')
pause
h=findobj(0,'tag','ejes2');
axes(h)
hold on
for i=1:length(xin)
  [y,t]=initial(modelo,xin(:,i));
  plot(t,y(:,1));
  pause(1)
end
grid on
disp('Pulse una tecla')
pause
h=findobj(0,'tag','ejes3');
axes(h)
hold on
for i=1:length(xin)
  [y,t]=initial(modelo,xin(:,i));
  plot(t,y(:,2));
  pause(1)
end
grid on
disp('Pulse una tecla')
pause
```

REFERENCIAS

García de Jalón, J., J. Ignacio y A. Brazales (1999). Aprenda Matlab 5.3. Universidad de Navarra, San Sebastián.

Roca, C.(199). Control de Procesos. Alfaomega.

Creus, S. A. (1998). Instrumentación Industrial, Alfaomega.

Hassan K. (1996) Nonlinear Systems, Prentice Hall.

EDITADO EN LOS TALLERES GRÁFICOS DE:

TECNOLOGIA OFFSET E.I.R.L.

Av. Bolivia 721 Breña
Telefax: 425-0799
Lima 05 - Perú