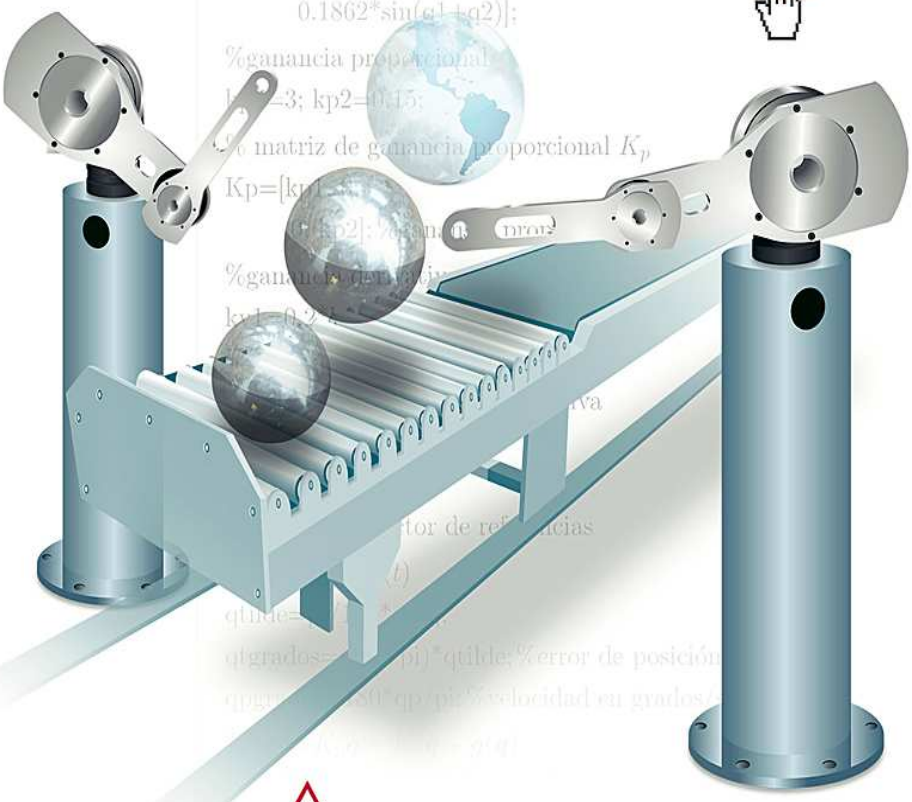
MECATRÓNICA

CONTROL Y AUTOMATIZACIÓN

FERNANDO REYES CORTÉS
JAIME CID MONJARAZ
EMILIO VARGAS SOTO

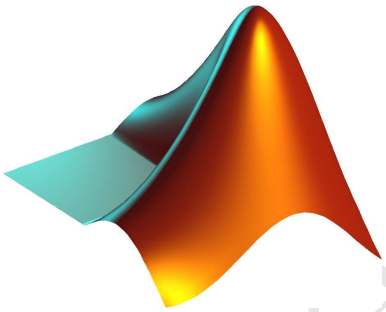
Contenidos Interactivos Web



Alfaomega

Capítulo 2

MATLAB para mecatrónica



Material Web

2.1 Simulink

SIMULINK es un ambiente interactivo de programación de bloques para modelar y simular una gran variedad de sistemas dinámicos y estáticos con estructura lineal y no lineal. Se trata de un entorno gráfico que representa al sistema como una interconexión de bloques elementales, cada bloque lleva asociado un modelo matemático que incluye la relación entrada/salida; combina potencia de programación y facilidad de uso del paquete de aplicación con flexibilidad y extensibilidad de un lenguaje de programación en bloques. Simulink permite seleccionar, adaptar y crear componentes de software y hardware para facilitar diversas aplicaciones con necesidades específicas.

2.1.1 Entorno gráfico de Simulink

Simulink proporciona un entorno gráfico al usuario que facilita el análisis, diseño y simulación de sistemas, incluye una serie de rutinas que resuelven cálculos matemáticos de fondo, junto con una interfase sencilla para programación de diagrama de bloques. Simulink usa diagramas interconectados de bloque para representar sistemas dinámicos y estáticos, posteriormente son compilados y ejecutados para llevar a cabo la simulación del proceso. Al iniciar Simulink, se obtiene una ventana como la que se muestra en la figura 2.1 que corresponde al navegador de librerías *ibrary browser*) de Simulink.

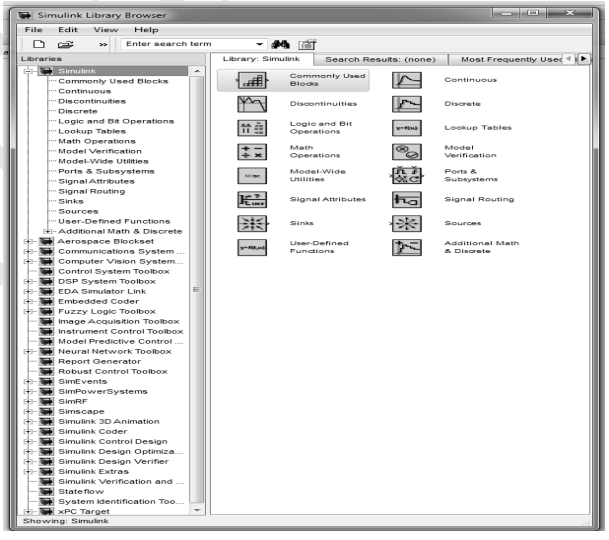


Figura 2.1 Sesión de inicio del ambiente integrado de Simulink.

En esta ventana se muestran todas las librerías disponibles; cada una de ellas contiene bloques elementales con los que se pueden construir sistemas. Las librerías de bloques estándar están organizadas como subsistemas. Para utilizar el ambiente gráfico de programación de Simulink existen diferentes modos para iniciar una sesión de trabajo, se puede escribir Simulink desde la ventana de comandos de MATLAB. También es posible acceder al programa desde el botón Simulink ubicado en la barra de botones de MATLAB o simplemente abrir un nuevo archivo con la extensión `mdl`.

2.1.2 Bibliotecas de Simulink

Las librerías de Simulink (*Simulink Library Browser*) contiene un conjunto de componentes como fuentes de señales y voltaje, funciones de perturbación, dispositivos para la presentación y desplegado de datos y resultados, sistemas lineales y no lineales, conectores para flujo de señal, medidores de señales como osciloscopios, así como herramientas para crear nuevos bloques requeridos por el usuario. Las componentes básicas del Simulink se ilustran en la la figura 2.2.

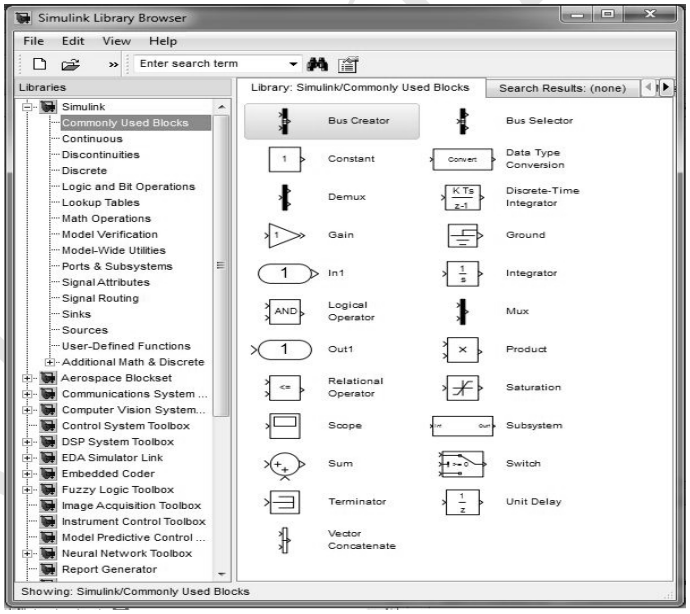


Figura 2.2 Navegador de librerías más comunes

La ventana de librerías de Simulink permite seleccionar los componentes básicos de modelado para realizar simulación en base a diagramas de programación con bloques interconectados simulación. En las ventanas de

modelo se dibujan los diagramas de bloques para realizar simulación del sistema a estudiar o analizar. Estas ventanas aparecen cuando se abre un modelo ya existente o se crea una ventana o plantilla en blanco para dibujar nuevos modelos. Para esto, se pueden utilizar los botones de la ventana de la librería de Simulink.

A continuación se describen algunos de los componentes básicos más utilizados para implementar simulación de sistemas:

Bus Creator: genera de un bus de las señales.

Bus Selector: selecciona las señales de bus de entrada.

Constant: proporciona una señal de valor constante.

Data Type Conversion: convertir la señal de entrada al tipo de datos especificado.

Demux: permite la descomposición de los datos puestos en forma vectorial en una línea mediante un multiplexador. Parámetro número de salidas.

Gain: aplica una ganancia constante a la entrada.

In1: por defecto un subsistema no tiene entradas. Por cada entrada que se debe de añadir se debe incluir uno de estos bloques.

Integrator: la salida del bloque se corresponde a la integral de la entrada. Los parámetros del bloque permiten controlar el valor inicial de la salida, así como la existencia de límites superiores e inferiores en la salida.

Mux: permite la inclusión de un conjunto de señales en una única línea de transmisión (que transmite datos vectoriales), lo que facilita la representación en el diagrama o dibujo. Parámetro número de entradas. Admite tanto entradas escalares como vectoriales.

Out1: por cada salida que se debe añadir se debe incluir uno de estos bloques.

Product: calcula el producto escalar de sus entradas. Un parámetro del bloque permite regular el número de entradas al mismo.

Saturation: la señal de salida no sobrepasa un valor umbral, configurable en los parámetros del bloque.

Scope: representa gráficamente la evolución en el tiempo de una variable o señal.

Subsystem: permite la realización de sistemas jerárquicos. Al abrir el subsistema, permite incluir en su interior, nuevos bloques constructivos e incluso anidar nuevos subsistemas.

Sum: calcula la suma de todas las entradas. Un parámetro permite indicar el número de entradas y si estas deben invertirse antes de la suma. Ejemplo: un valor para el parámetro `++-+` indicaría que el bloque tiene 4 entradas y la tercera de ellas se invierte antes de sumarla.

Switch: una entrada del sistema permite

seleccionar cual de las otras dos entradas se presenta en la salida.

Continuos: modelos que representan funciones continuas en el tiempo.

Derivative: la salida del bloque se corresponde a la derivada de la entrada.

Transfer Fcn: permite expresar una función de transferencia en el dominio de la variable compleja s. Sus parámetros son los polinomios del numerador y del denominador de la función de transferencia, expresados como vectores filas.

Transport Delay: la salida del bloque corresponde con la entrada al mismo retrasada una cantidad de tiempo, que se fija como parámetro en el bloque.

Zero-pole: Función de transferencia expresada en función de la ganancia en régimen permanente, y la situación de los polos y ceros del sistema.

Math Operations: bloques que realizan operaciones matemáticas sobre sus entradas.

Abs: calcula el valor absoluto de su entrada.

Math Function: este bloque incluye la mayor parte de las funciones matemáticas típicas, con la excepción de las funciones trigonométricas.

Sign: calcula el signo de la entrada +1 indica positivo, -1 indica negativo y 0 un valor nulo.

Trigonometric Function: en este bloque se incluyen todas las funciones trigonométricas típicas.

Discontinuities: Bloques con funciones discontinuas.

Dead Zone: incluye una zona muerta en el sistema, centrada en torno al cero. El sistema no responde ante estos valores. La magnitud de la zona muerta pude modificarse y hacerse simétrica por medio de los parámetros del sistema.

Relay: la salida pasa a un estado ON=1 cuando la entrada supera un valor umbral y OFF=0 cuando se encuentra por debajo de un umbral distinto. El estado inicial es OFF.

Signals Routing: manejo de sistemas y señales.

Data Store Memory: define una variable del entorno de trabajo que se va a utilizar como lugar de almacenamiento de datos útiles para evitar realizar conexiones complejas que compliquen el diagrama de bloques que se está implementando.

Sinks: sumideros de señales.

Display: representa numéricamente el valor de una variable o señal.

To Workspace: guarda el valor de la señal indicada en una variable del entorno de trabajo del Matlab. Se puede asignar el nombre de la variable y limitar su tamaño.

To File: guarda en n fichero tipo .mat los datos de la señal de entrada a este bloque.

Stop Simulation: detiene la simulación si el valor de la entrada es distinto de Cero.

Sources: Fuentes de señales.

Chirp Signal: genera una señal senoidal, modulada en frecuencia entre un valor inicial y final.

Clock: tiempo se lleva la simulación.

From Workspace: proporciona una secuencia de datos tomadas del entorno de trabajo de Matlab. La variable seleccionada debe contener una matriz indicando los valores de la señal y los instantes de tiempo en los que la señal toma cada valor.

From File: proporciona datos tomados de un fichero tipo `.m` en el que debe estar el valor de la variable, junto a los instantes de tiempo en que toma cada valor.

Pulse generator: genera una onda cuadrada, o pulso rectangular, de la que se puede controlar la amplitud, el período que dura esta amplitud y la relación entre el tiempo que la onda toma su valor máximo y el tiempo que toma el valor mínimo.

Ramp: genera una señal tipo rampa, o función lineal de pendiente constante.

Random Number: genera números aleatorios distribuidos normalmente.

Signal Generator: simula un generador de señales electrónico, permitiendo generar ondas senoidales, dientes de sierra, ondas cuadradas o aleatorias.

Sine Wave: generador de ondas senoidales.

Step: genera una señal tipo escalón.

Insertar y pegar bloques

Para generar un nuevo modelo se pulsa el botón nuevo modelo, apareciendo el espacio de trabajo de un modelo como se muestra en la figura 2.3.

Buscar un bloque: Se puede buscar un bloque expandiendo el árbol de la biblioteca o buscándolo directamente por su nombre en la ventana de búsqueda. En este caso, si hay más de un bloque que pueda corresponder a ese nombre, irán apareciendo a medida que se pulse la tecla `enter`.

Situar un bloque: Para situar un bloque, se mantiene pulsado el botón izquierdo del ratón sobre el icono en forma de rombo que hay junto al nombre del bloque y se arrastra hacia la posición deseada en la ventana de simulación como se presenta en la figura 2.4.

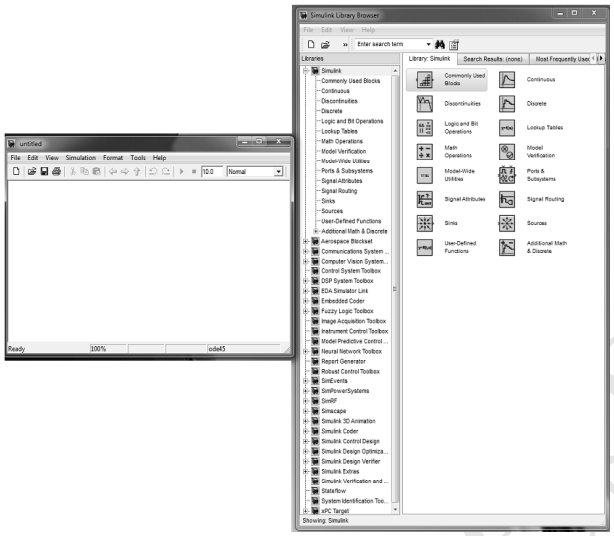


Figura 2.3 Nuevo modelo

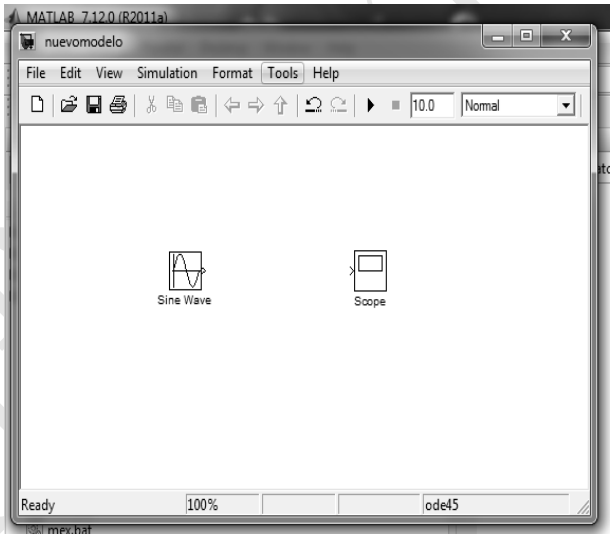


Figura 2.4 Insertando bloques

Conectar bloques: En cada bloque, los puntos de salida aparecen indicados mediante una flecha saliente del bloque, mientras que los puertos de entrada a cada bloque se indican con una flecha entrante al mismo. Se conecta la entrada de un bloque a la salida de otro, manteniendo pulsado el botón izquierdo del ratón mientras se arrastra desde el símbolo de entrada de uno de los bloques hasta el de salida de otro o viceversa como se muestra en la figura 2.5.

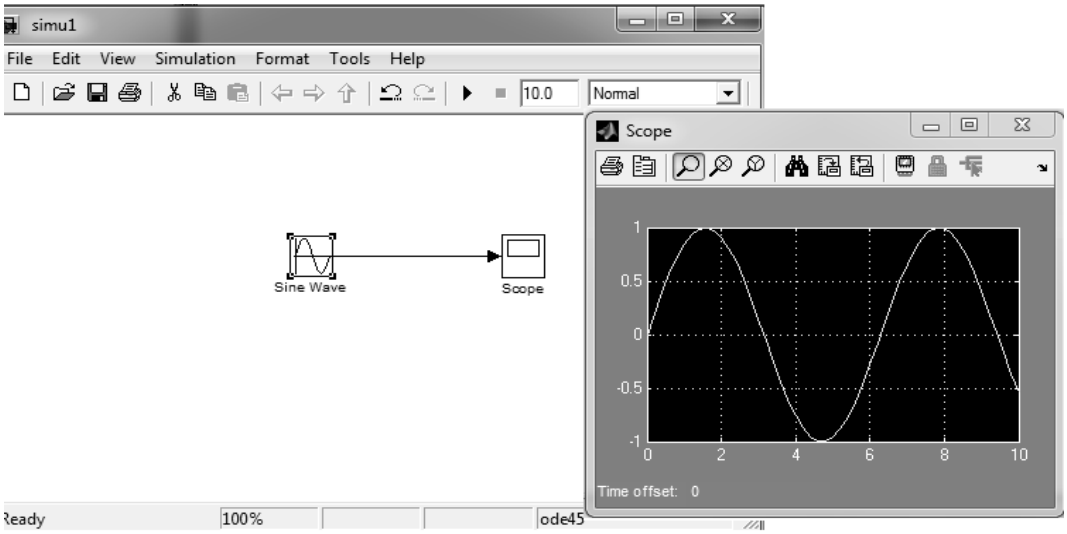


Figura 2.5 Conectando bloques y uso del osciloscopio

Un ejemplo básico incluiría la selección de dos componentes `Simulink \Sources \Sine Wave` y `Simulink\Sinks\Scope` de la ventana (*Simulink Library Browser*), y el arrastre de los mismos hasta la ventana de dibujo. En el caso de este ejemplo básico, para conectar el generador de señales y el osciloscopio, simplemente se debe situar el ratón sobre el punto de salida del generador, pulsar el botón izquierdo, arrastrar el ratón hasta el punto de entrada del osciloscopio y soltar el botón del ratón. Este programa será guardado con el nombre `simu1.mdl`; pulsar el botón de `play` para realizar la simulación del programa como se muestra en la figura 2.5.

Bifurcaciones

Para llevar la salida de un bloque a la entrada de más de un elemento se necesita generar una bifurcación en la conexión. Para hacerlo, se arrastra con el ratón desde la entrada del nuevo bloque a conectar hasta la línea de la conexión que se va a bifurcar.

Modificar bloques

Se pueden rotar o aplicar simetrías para modificar a los bloques usados, según convenga la colocación de entradas/salidas para el esquema que se esté realizando pulsando sobre él el botón derecho del ratón y utilizando los menús desplegables o mediante la opción *Format* del menú principal *(Format Flip Block, Format Rotate Block, etc)*. También mediante los menús o haciendo doble *clic* sobre el bloque, se pueden

modificar sus parámetros.

Insertar textos

Se puede incluir un texto aclaratorio o informativo en cualquier parte de la ventana del modelo, haciendo doble clic en una zona libre y escribiendo directamente el texto. También se pueden cambiar los nombres y posiciones de los bloques que se emplean para la simulación antes o después de conectarlos. Así mismo los enlaces de las conexiones pueden moverse o modificarse. Para eliminar cualquier elemento basta con seleccionarlo con un *clic* y eliminarlo con la tecla `Supr` o `delete`, o utilizar alguno de los menús.

♣ Ejemplo 2.1

Realizar un programa en diagrama de bloques para Simulink que obtenga la derivada de una señal senoidal.

Solución

Para realizar este problema, primero es conveniente realizar una copia del archivo `simu1.mdl` (ver figura 2.5) como `simu2.mdl` usando la opción (`File  Save as`).

Ahora se coloca un bloque derivador de la señal senoidal y un bloque *bus creator* como se presenta en la figura 2.6, se realiza un *click* en el botón `play` para realizar la simulación.

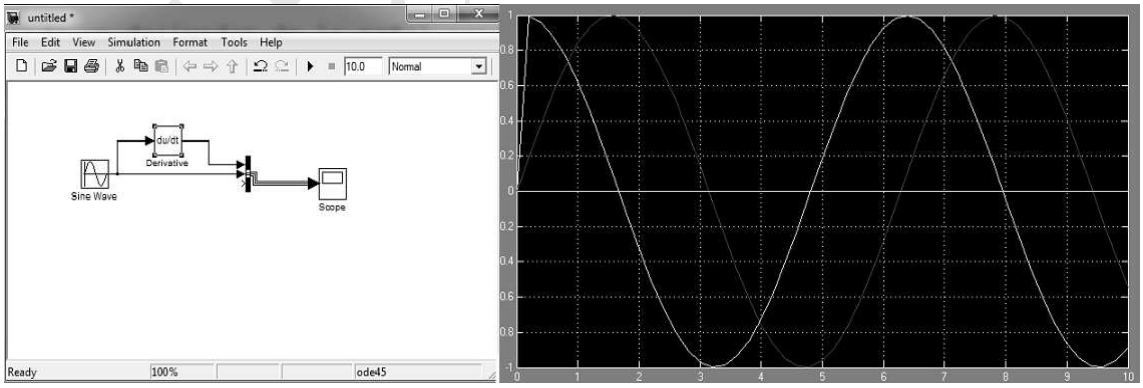


Figura 2.6 Modelo con un derivador de una señal senoidal.

♣ Ejemplo 2.2

Realizar un programa en diagrama de bloques para Simulink que obtenga la integral y derivada de una señal senoidal.

Solución

Renombrando el archivo `simu2.mdl` como `simu3.mdl` se inserta un bloque integrador como se presenta en la figura 2.7 (pulsar `play`).

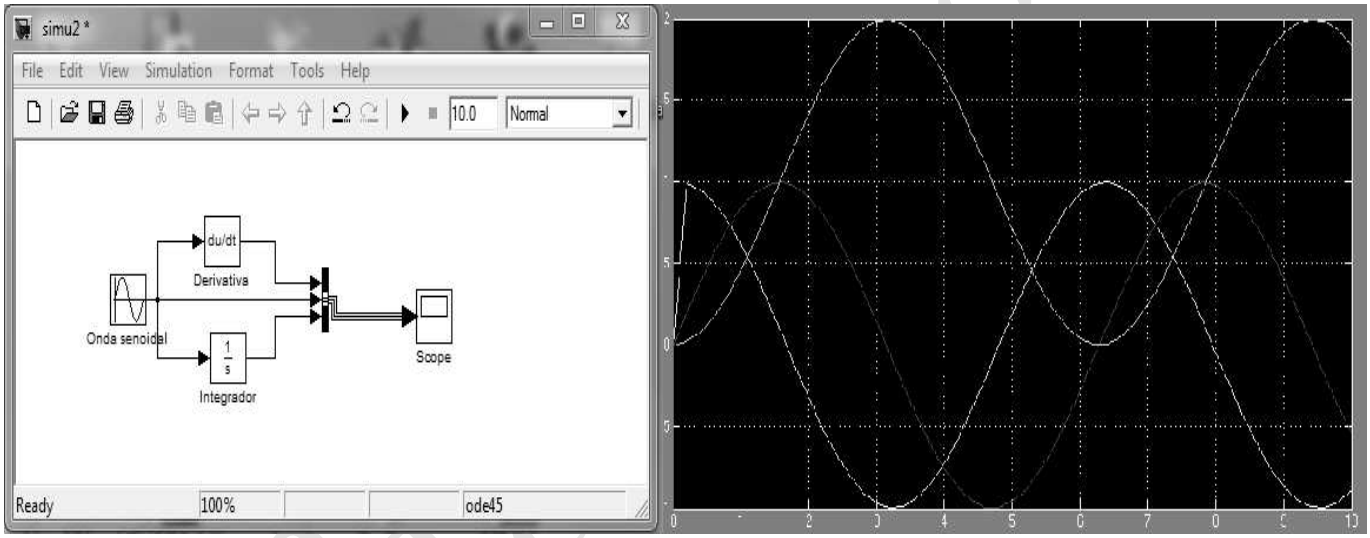


Figura 2.7 Modelo con un derivador e integrador de la señal senoidal.

■■■

♣♣ Ejemplo 2.3

Realizar un programa en diagrama de bloques que reproduzca el audio de la suma de tres señales senoidales y su desplegado en un osciloscopio.

Solución

La figura 2.8 muestra el programa en diagrama de bloques `simu4.mdl` el cual contiene tres ondas senoidales con diferente frecuencia conectadas a un bloque sumador cuya la salida es conectada a una bocina para

reproducir el audio de la composición de las tres señales senoidales.

También se despliegan en un bloque osciloscopio las formas de cada una de las señales que producen el sonido en el elemento acústico.

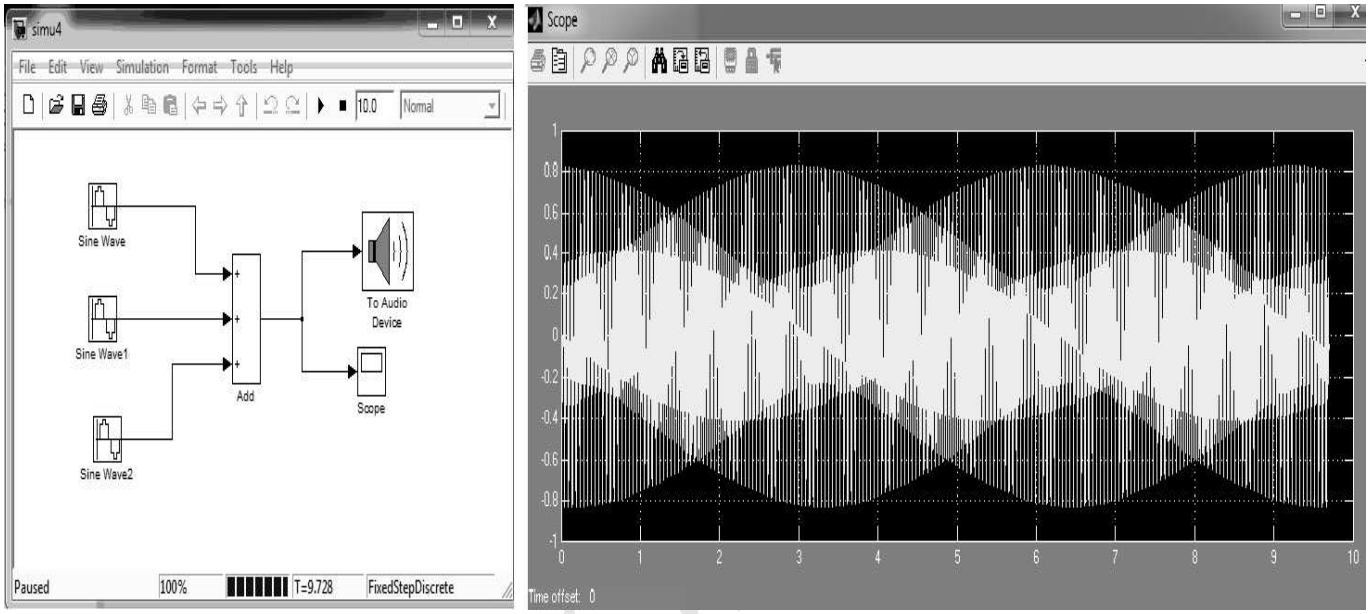


Figura 2.8 Modelo con tres señales senoidales con distinta frecuencia conectadas a un bloque de audio.

■■■

♣♣ Ejemplo 2.4

Muestrear la respuesta a un escalón de una función de transferencia de segundo orden $\frac{1}{s^2+ss+1}$ con un retenedor de orden cero, cuyo periodo de muestreo es de 0.2 segundos.

Solución

La figura 2.9 muestra el programa `simu5.mdl` contiene una señal con entrada escalón unitario a la función de transferencia de segundo orden en tiempo continuo cuya salida es muestreada por un retenedor de orden cero con periodo de muestreo de 0.2 segundos.

La gráfica de la simulación con este retenedor de orden cero se puede observar en lado derecho de la figura 2.9.

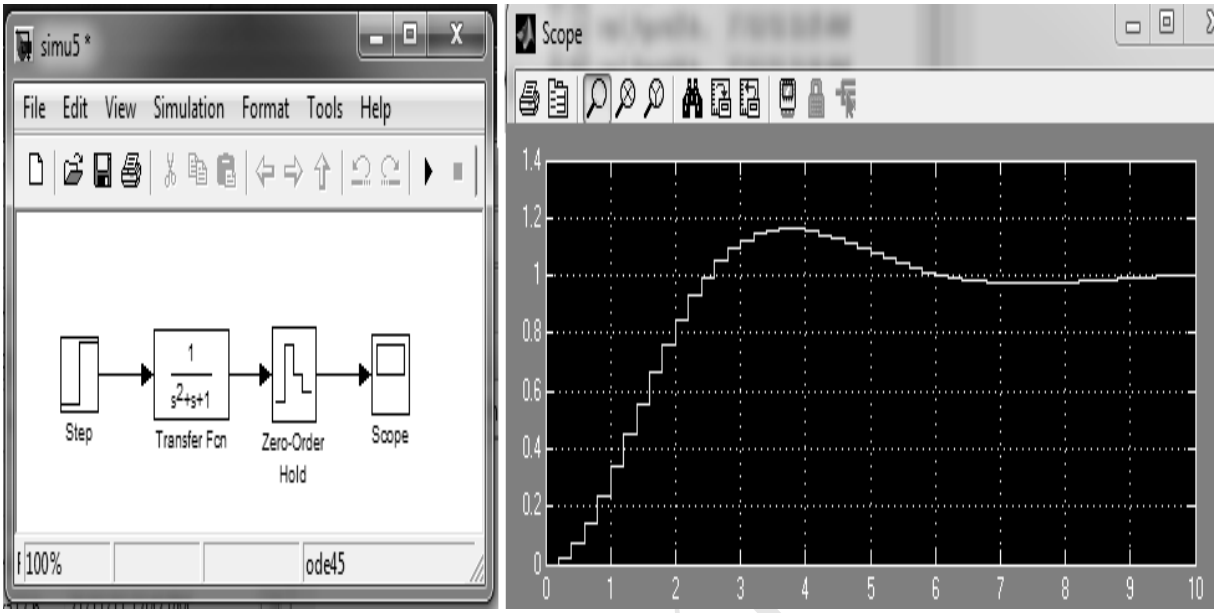


Figura 2.9 Modelo de función de transferencia en lazo abierto

■■■

♣♣♣ Ejemplo 2.5

Realizar un programa en Simulink que implemente la siguiente ecuación diferencial:

$$\frac{d}{dt}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 0 & 2 \\ -0.5 & -0.1 \end{bmatrix}\begin{bmatrix} x_1 \\ x2 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u$$

donde la señal u es una función escalón unitario.

Solución

En la parte superior de la figura 2.10 se muestra el programa `simu6.mdl` con la programación en diagrama de bloques interconectados que implementa a la ecuación diferencial solicitada.

En la programación de los diagrama de bloques se emplean bloques de amplificadores operacionales para implementar las ganancias de las constantes. Además, también se hace uso de dos bloques integradores. La aceleración $\dot{x}_2$ pasa por el primer integrador para obtener la velocidad x_2 la cual pasa por el segundo integrador para obtener la posición x_1. Se emplea un bloque *bus creator* para desplegar las señales del sistema

sobre un bloque medidor tipo osciloscopio.

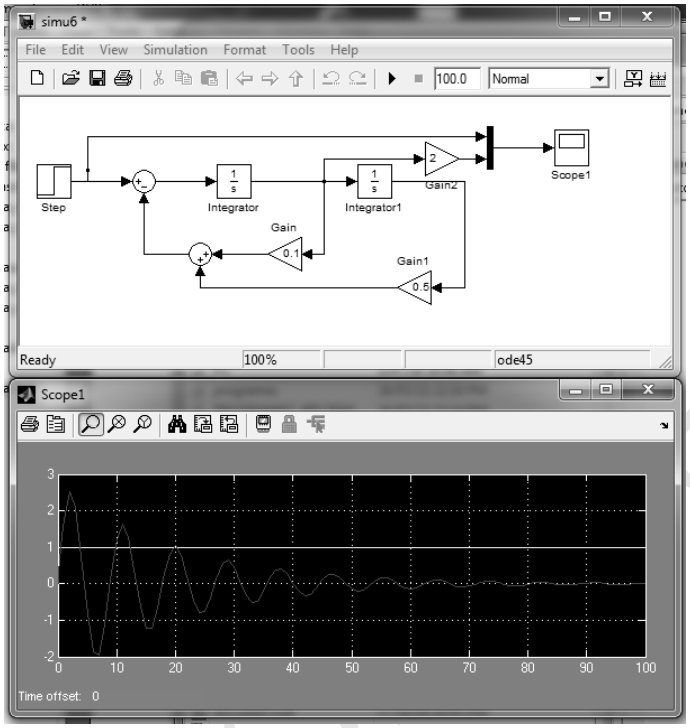


Figura 2.10 Modelo de ecuación diferencial y respuesta a un escalón unitario.

■■■

♣♣♣ Ejemplo 2.6

Realizar un programa en Simulink que controle la posición de un motor de DC.

Solución

El programa que controla la posición de un motor de corriente directa se denomina *simu7.mdl* y se describe en la figura 2.11. Este programa utiliza bloques básicos de control proporcional integral (PI) para la regulación de velocidad y par o torque aplicado al motor. También se emplea un bloque de motor de DC el cual admite como entrada el voltaje de armadura y torque.

La salida que produce es la posición del motor, la corriente de la armadura y la señal de frecuencia que está directamente relacionada con la velocidad de movimiento del motor.

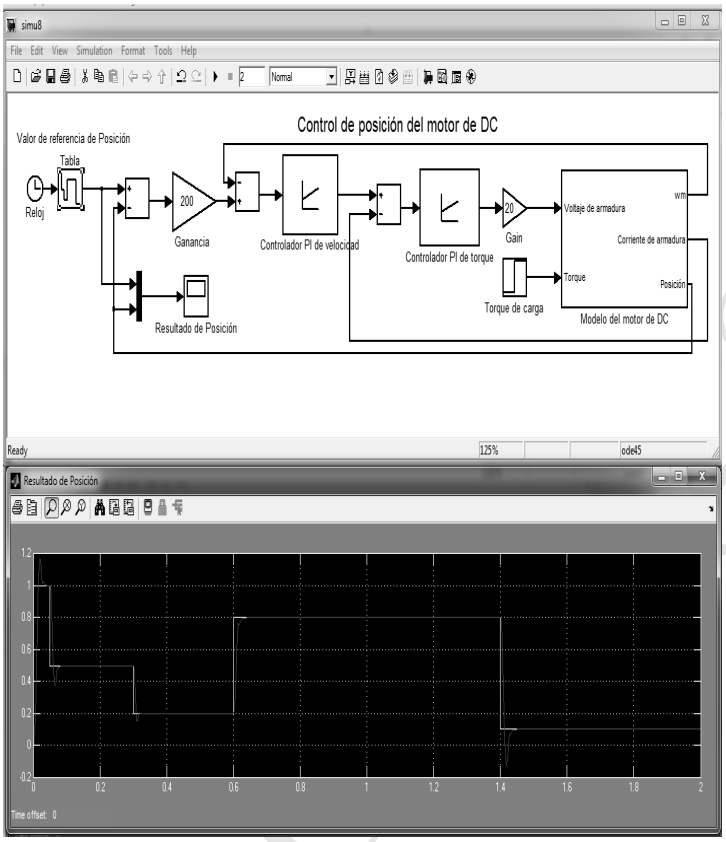
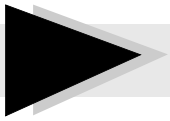


Figura 2.11 Modelo de control de posición de un motor de DC.



2.2 Interfaz Gráfica de Usuario

La interfaz gráfica de usuario de **MATLAB** recibe el nombre de *Graphical User Interface Development Enviroment* (**GUIDE**), es la manera en que el usuario puede interactuar con un programa de forma sencilla, de tal forma que permite trabajar intuitivamente. Contiene diferentes elementos gráficos tales como: botones, barras deslizantes, campo de texto, menús, gráficos, inspector de propiedades, etc.

2.2.1 Entorno GUIDE

GUIDE proporciona un entorno gráfico al usuario que facilita el diseño de aplicaciones por medio de bloques para representar enlaces dinámicos generando una interfaz gráfica al usuario para interactuar con el programa.

Existen varias formar para iniciar una sesión de trabajo con `GUIDE`; una posible puede ser empleando la ventana de comandos por teclear:

fx >> `GUIDE`

Otra forma puede ser por acceder al programa desde el botón `GUIDE` en la barra de botones del entorno de **MATLAB** o simplemente por abrir un archivo con extensión `gui`.

Cuando se inicia el ambiente GUIDE se obtiene una ventana como la que se muestra en la figura 2.12, hay dos opciones de trabajo sobre esta ventana inicial: una puede ser para crear una nueva aplicación GUI o abrir alguna ya existente.

Para crear una nueva aplicación, se deben tomar en cuenta las siguientes opciones:

Interfaz gráfica en blanco

Presenta una interfaz gráfica en blanco (*Blank GUI*) la cual viene en forma predeterminada con un formulario nuevo, en el cual se puede diseñar el programa como se ilustra en la figura 2.13. Observe la paleta de componentes disponibles; cada una de ellas contiene bloques elementales con los que se pueden construir la interfaz. Cada uno de estos elementos tienen un conjunto de propiedades en las cuales se puede acceder con

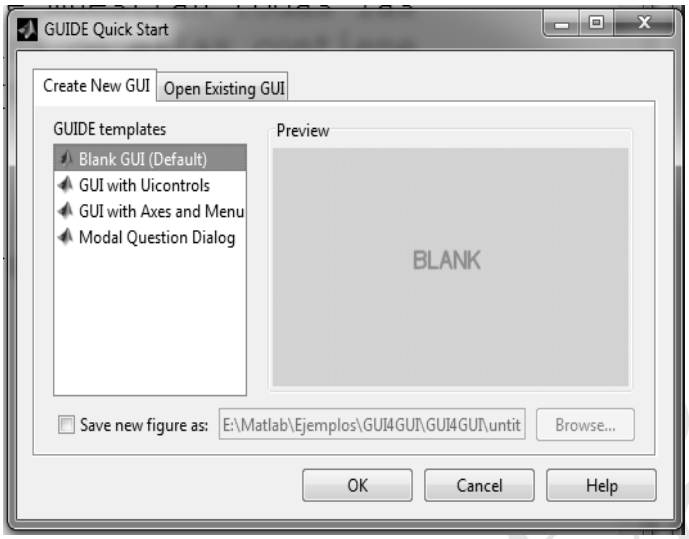


Figura 2.12 Inicio del entorno GUIDE.

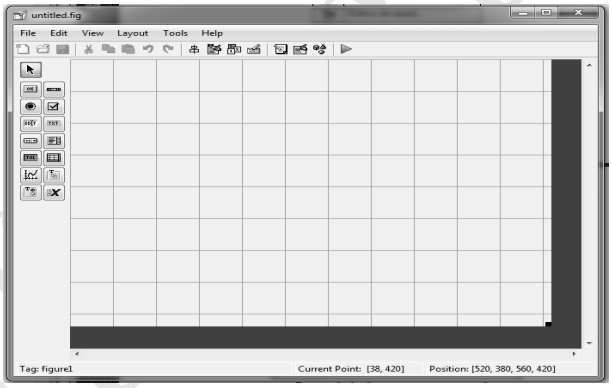


Figura 2.13 Sesión nueva en blanco.

el botón derecho del *mouse*, una vez pulsado este aparece la siguiente imagen como lo indica la figura 2.14:

Para editar las propiedades de cada elemento se selecciona la opción *Properties Inspector* y se abre una consola, la cual variará según que elemento que se este editando. Las propiedades que se pueden editar son: color, posición, tamaño, tipo de letra, etc. como se muestra en la figura 2.15:

También es posible insertar menús con opciones para abrir, cerrar e imprimir archivo. El formulario tiene menús tipo *popup menu*, un botón para pulsar (*push button*) y un objeto para realizar ejes gráficos.

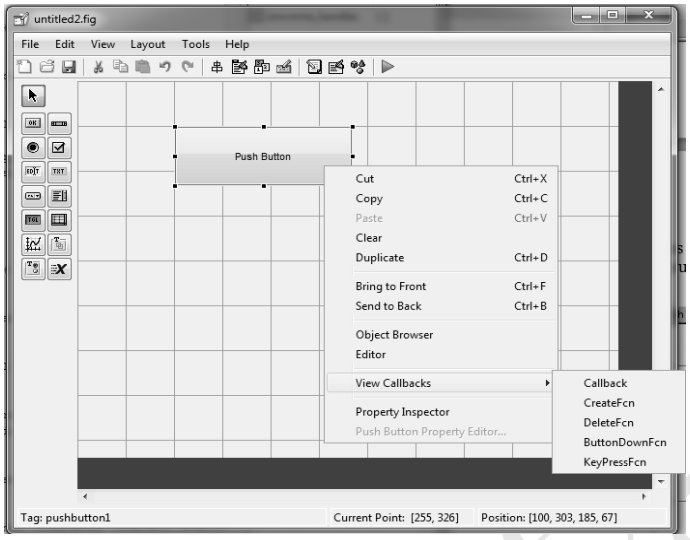


Figura 2.14 Propiedades del elemento.

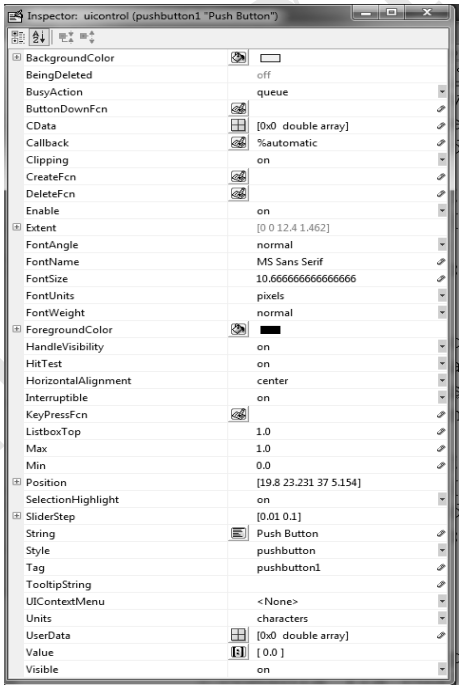


Figura 2.15 Editor de inspector de propiedades.

Las aplicaciones realizadas en `GUIDE` constan de dos archivos, uno de ellos es ejecutable en MATLAB con extensión `.m`; tiene una estructura predeterminada con encabezado y código correspondiente de las funciones o subrutinas para realizar las actividades programadas. El segundo archivo es la parte gráfica con extensión `.fig`; ambas partes están unidas a través de las subrutinas *callback*. Una vez que se graba la aplicación `.fig` desde del entorno `GUIDE`, automáticamente también lo hace para el archivo asociado `.m` ; la aplicación se puede ejecutar en la ventana de comando de MATLAB escribiendo el nombre del archivo que tiene extensión `.m`; por ejemplo, si se guarda con el nombre `gui1.fig` de manera automática se graba el archivo `gui1.m`. En la ventana de comando se puede teclear

fx >> `gui1`

se ejecuta el programa de la figura 2.16.

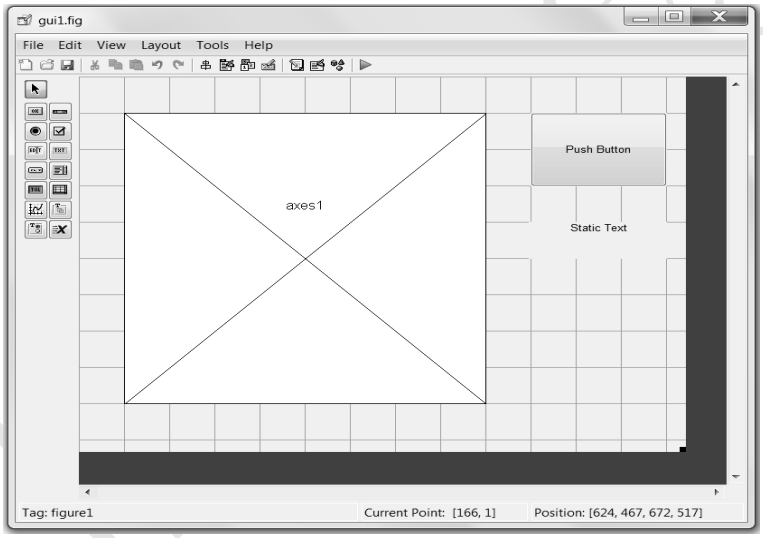


Figura 2.16 Ejemplo de una interfaz gráfica de usuario.

Pulsando el botón derecho en cada uno de los iconos que utiliza GUIDE se llama al inspector de propiedades y se puede cambiar los valores contenidos, por ejemplo el nombre del *Static text*, no posee una función asociada, pero sí una dirección, que se puede utilizar para escribir comentarios. Para saber cuál es esta dirección, también se puede hacer doble *click* en esta componente, y se ubica la etiqueta *Tag y String* como se muestra en la figura 2.17:

La figura 2.18 presenta una interfaz gráfica donde el usuario introduce una serie de datos de un sistema

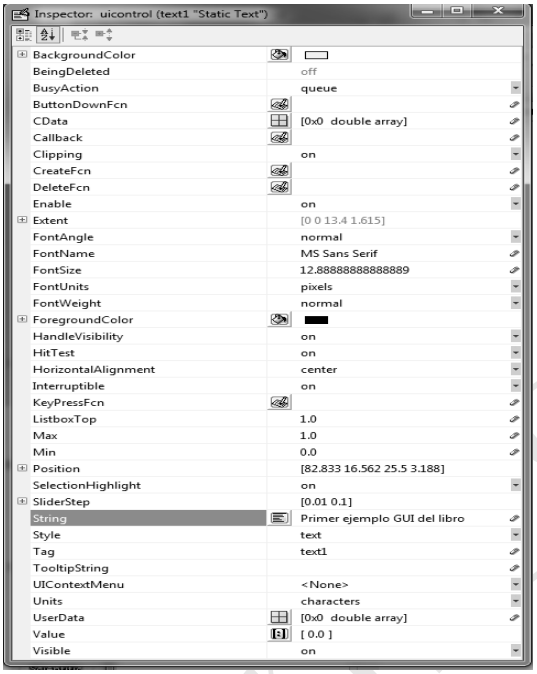


Figura 2.17 Inspector de Propiedades para el componente Static Text.

masa resorte amortiguador sencillo.

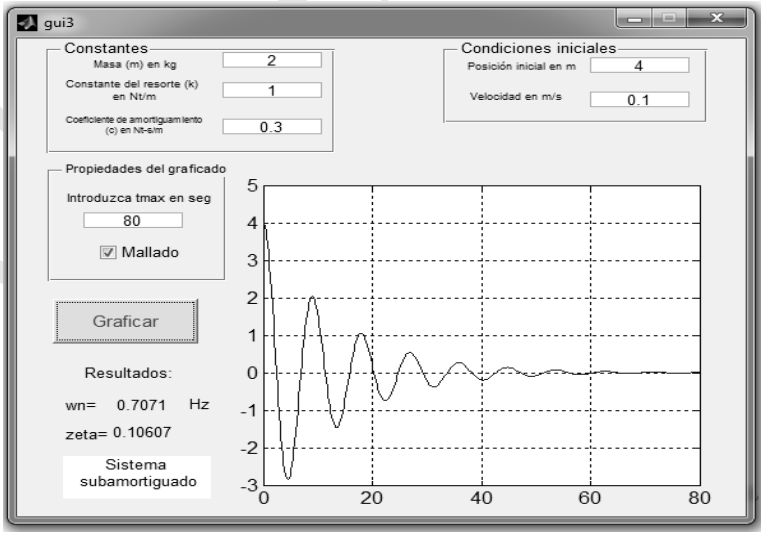


Figura 2.18 Programa GUI de sistema masa, resorte y amortiguador