

MATLAB 图形用户界面（GUI）设计基础

图形用户界面（Graphical User Interface, GUI）是包含图形对象（例如窗口、图标、菜单、文本等）的用户界面，它是应用程序中很重要的一部分，为用户和计算机或计算机程序提供了通信、交互的平台，是用户与计算机程序之间的交互方式。通过图像用户界面，用户可以轻松直观地和计算机交换信息，用户不需要了解应用程序是如何执行每一条命令的，只需要掌握图形界面各个组件的操作方法即可，对于使用应用程序的用户而言，界面就是应用程序，他们感觉不到正在幕后运行的代码。

1. GUI 对象层次与设计原则

GUI 由各种类型的图形对象组成，在 MATLAB 中，基本的图形对象有窗口（Figure）、菜单（Uimenu）、控件对象（Uicontrol）、坐标轴对象（Axes）等，它们的层次关系如图 1 所示。每个图形对象都有一个唯一的 ID，称为句柄（handle），因此 MATLAB 的图形对象也称为句柄图形对象，用户可以通过句柄查询或设置图形对象。

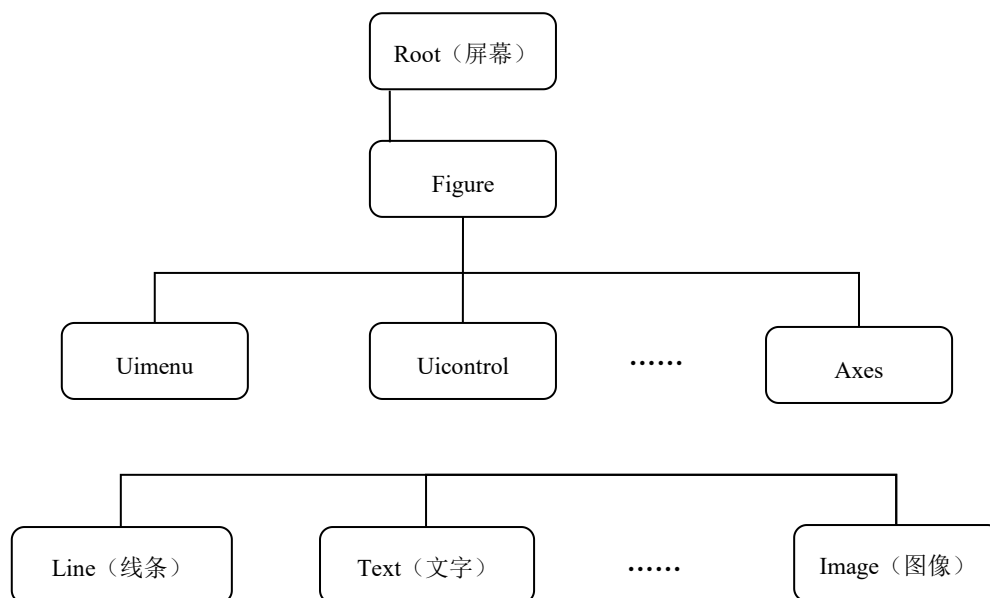


图 1 图形对象结构层次关系图

简单性(Simplicity)、一致性(Consistency)和熟悉性(Familiarity)是用户界面设计的三大原则。

简单性：设计界面时，要力求简洁、直接、清晰的表现界面的功能和特征。在界

面上，只提供那些引导用户完成实际工作必需的功能，保持界面整洁，要尽量减少窗口的数量，一些相关的功能最好能集中在一个窗口里，避免过多的在不同窗口之间切换。

一致性：尽量使一个程序的界面和其他程序的界面风格保持一致，或者程序的多个功能类似的窗口风格保持一致，保持了一致性会使用户更方便的使用你所开发的程序界面。

熟悉性：尽量使用人们所熟悉的标志和符号，有时候虽然不知道该怎么操作，但根据图形的形状等可以做出有效的猜测。

2.GUI 编辑器

MATLAB 提供了 GUI 设计工具 GUIDE。从命令窗口中输入 `guide` 命令，或者从【File 菜单】→New→GUI 可以启动 GUIDE 进入 GUI 界面编辑环境，如图 2 所示。

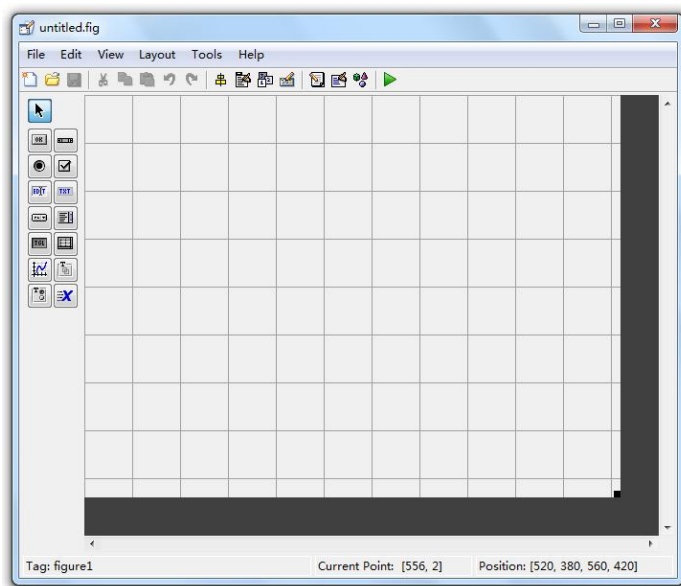


图 2 GUIDE 编辑器

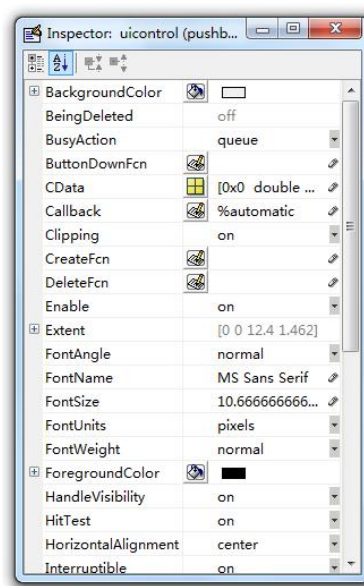










图 3 属性编辑器

窗口左侧是对象工具栏，包含了各种常用对象，包括：按钮（Push Button）、滚动条（Slider）、单选按钮（Radio Button）、复选框（Check Box）、文本编辑框（Edit Text）、静态文本框（Static Text）、下拉菜单（Pup-up Menu）、列表框（Listbox）、开关按钮（Toggle Button）、表格（Table）、坐标轴（Axes）、组合面板（Panel）、组合按钮（Button Group）、ActiveX 控件（ActiveX Control）等。

从对象工具栏选中控件拖动到右侧的界面布局区即可创建对象。选中创建的对象后点击上方快捷工具栏中的按钮可以打开属性编辑器（如图 3 所示），即可通过参数设置对象的属性。未选中对象时点击可编辑窗口的属性。

快捷工具栏中还提供了其他工具：点击按钮可以开启菜单编辑器（图 4）；点击按钮可以开启工具栏编辑器（图 5）；点击按钮打开代码编辑器；点击按钮打开对象浏览器；点击可以执行程序。

我们试着建立一个简单的界面，首先拖动一个坐标轴和一个按钮到界面布局区（图 6），然后分别设计一个菜单（图 4）和工具栏（图 5）。然后保存文件，图形用户界面文件保存为“.fig”格式的文件，同时会生成与之对应的一个“.m”文件。点击执行程序，程序运行界面如图 7 所示。

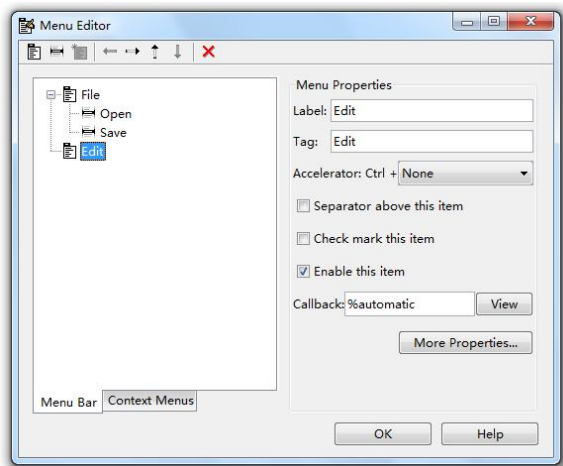


图 4 菜单编辑器图

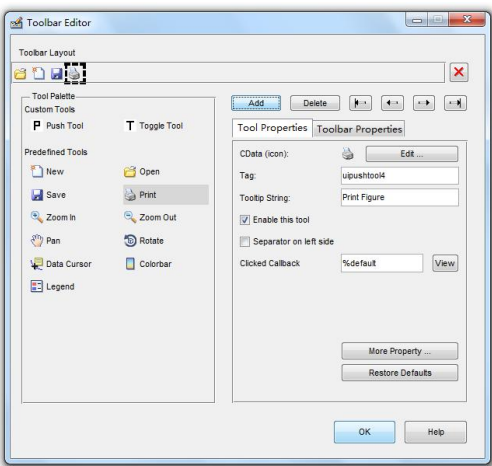


图 5 工具栏编辑器

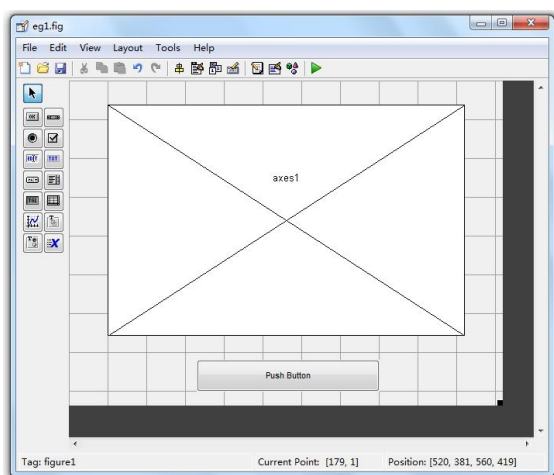


图 6 创建控件

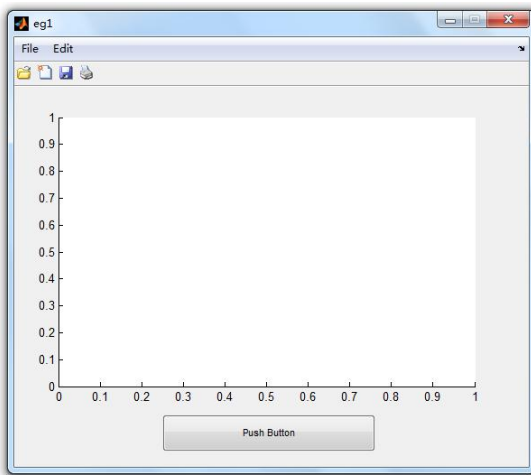




图 7 程序运行界面

3. 菜单

通过菜单编辑器（图 8）可以设计两种菜单，菜单编辑器左下角可以选择建立菜单的类型 **Menu Bar**（一般图形菜单）和 **Context Menus**（右键菜单）。单击右上角工具栏中的按钮  可以建立主菜单，选定任意一个主菜单单击  按钮可以建立该主菜单的选项。菜单编辑器右侧的“Menu Properties”可以设置菜单的基本属性。

Label: 菜单显示的名称，如 File。可以采字符串中添加“&”符号设置快捷键，例如 &File 表示 & 后方的第一个字母 F 是此菜单的快捷键，执行这个程序后，该菜单会显示为 File，当按下 Alt+F 组合键时会弹出该菜单。

Tag: 用户指定的对象标记，当用户借助 findobj 函数时，可以根据这个 Tag 属性值找出对应的句柄。

Accelerator Ctrl+: 指定快捷键，当用户按下 Ctrl+指定的快捷键时，就会执行该菜单的操作。

Separator above this item: 选中后在该菜单选项的上方会出现水平分隔线。

Check mark this item: 选中后，选取该菜单选项会出现“√”标记。

Enable this item: 若未选中该选项，执行后该菜单选项会显示为灰色，且无法选取。

Callback: 设置选取该菜单选项时执行的操作。如果使用默认值，则会在保存用户界面文件时生成的“.m”文件中加入一个空的 Callback 函数，可以再该函数中添加选取该菜单选项时需要执行的代码，点击按钮“View”可以开启 M 文件编辑器进行编

辑。

More Properties: 打开属性编辑器，可以设置菜单更多的属性。

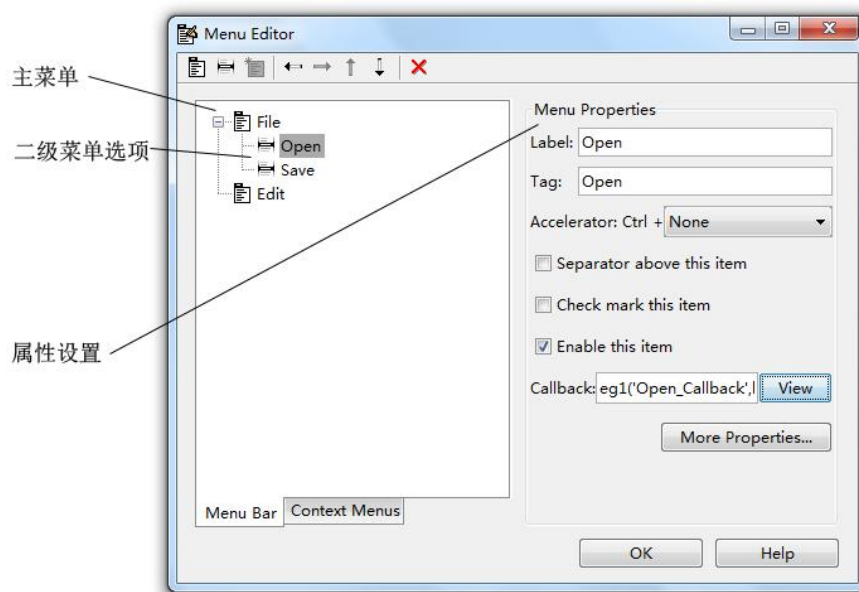



图 8 编辑菜单

4.GUI 对象

GUIDE 编辑器左侧的对象工具栏可以选择各种常用的对象，单击快捷工具栏中的按钮可以打开属性编辑器设置选中对象的属性。

(1) 按钮（Push Button）：最常用的对象，string 属性指定按钮上显示的内容，单击时执行的操作在其对应的 Callback 函数中设定，在 GUIDE 编辑器中右键点击按钮 Push Button 对象，在右键菜单中选自“View Callback”→“Callback”开启 M 文件编辑器进行编辑。

(2) 滚动条（Slider）：用于从一定的范围中取值。包括三个部分，分别是滚动槽，表示取值范围；滚动槽内的滑块，代表滚动条的当前值；以及在滚动条两端的箭头，用于改变滚动条的值。

(3) 单选按钮（Radio Button）：由一个字符串和字符串左侧的一个小圆圈组成。选中时，圆圈被填充一个黑点，且属性 Value 的值为 1；未选中时，圆圈为空，属性 Value 的值为 0。可与 Button Group 组合使用。

(4) 复选框（Check Box）：由一个字符串和字符串左侧的一个小方框所组成。

选中时在方框内添加“√”符号，Value 属性值设为 1；未选中时，Value 属性值设为 0。

(5) 文本编辑框 (Edit Text)：允许用户动态地编辑文本字符串或数字，其内容在属性 string 中。

(6) 静态文本框 (Static Text)：用来显示文本字符串，显示内容是由属性 string 所确定的。与 Edit Text 对象不同的是，其内容不能直接编辑，只能通过改变其属性 string 属性来改变。

(7) 下拉菜单 (Pup-up Menu)：提供互斥的一系列选项清单，用户可以选择其中的某一项。Value 属性值为所选选项的序号，在 string 属性中设置下拉菜单的选项字符串，不同的选项之间用“|”分隔。

(8) 列表框 (Listbox)：提供一个列表式的选项清单，并允许用户选择其中的一个或多个选项，一个或多个的模式由属性 Min 和 Max 控制。Value 属性的值为被选中选项的序号，同时也指示了选中选项的个数。

(9) 开关按钮 (Toggle Button)：和 Push Button 形状相同，区别在于它有两种状态“开”(按钮按下)和“关”(按钮弹起)，用鼠标单击按钮，它会从一种状态变成另一种状态，并执行相应的 Callback 函数。开关按钮“开”时，Value 属性的值为在 Max 属性中指定的值，“关”时，Value 属性的值为在 Min 属性中指定的值。

(10) 表格 (Table)：用于以表格的形式显示数据。

(11) 坐标轴 (Axes)：用于显示运算结果图形。

(12) 组合面板 (Panel)：可以包含其他的 GUI 对象，一般把一组密切相关的对象放在同一个 Panel 对象中。

(13) 组合按钮 (Button Group)：一般用于 Radio Button 或 Toggle Button 两类对象，用于管理互斥选取的行为。将一组 Radio Button 或 Toggle Button 对象放置在一个 Button Group 对象中，当选中其中一个 Radio Button 或 Toggle Button 对象，同组的其他对象将自动处于未选中状态。我们还可以结合 switch-case 语句来设定 Callback 函数，例如我们建立一个 Tag 名称为 uipanel 的 Button Group 对象，在其中建立两个 Radio Button 对象，Tag 名称分别为 radiobutton1 和 radiobutton2，在 Button Group 对象的 Callback 的 SelectionChangeFcn 函数中编写如下代码

```
function uipanel1_SelectionChangeFcn(hObject, eventdata, handles)
```

```

% hObject    handle to the selected object in uipanel2
% eventdata  structure with the following fields (see UIBUTTONGROUP)
%  EventName: string 'SelectionChanged' (read only)
%  OldValue: handle of the previously selected object or empty if none was selected
%  NewValue: handle of the currently selected object
% handles    structure with handles and user data (see GUIDATA)

selection = get(handles.uipanel1,'SelectedObject');
switch get(selection,'Tag')    %通过 Tag 属性判断选中的对象
case 'radiobutton1'
%选中 Tag 属性为 radiobutton1 的对象时执行的代码
.....
case 'radiobutton2'
%选中 Tag 属性为 radiobutton2 的对象时执行的代码
.....
end

```

（14）ActiveX 控件（ActiveX Control）：可以与其他软件接口有效结合，或直接操作外来的组件。

5.一些小技巧

（1）获取当前图形对象的句柄

MATLAB 定义了 3 个特别的函数来访问一些当前的图形对象，它们是

gcf 取得当前的图形窗口的句柄值

gca 取得当前的坐标系的句柄值

gco 取得当前的图形对象的句柄值

可以将这些函数取得的句柄值作为输入变量送到需要这些句柄值的函数中。

（2）图形对象属性操作

set 函数可以让用户设置图形对象的属性值，格式为

set(handle, '属性 1', '属性值 1', '属性 2', '属性值 2', ...)

它的第一个输入参数 `handle` 是图形对象的句柄值，例如 `set(gcf,'color',[1 1 1])`表示将当前图形窗口对象的颜色设为白色。

与 `set` 相对应，`get` 函数可以用来取得图形对象的各项属性的值。例如 `get(handle, 'position')` 返回句柄为 `handle` 的图形对象的位置属性。

`findobj` 函数返回具有指定属性值的所有对象，例如 `findobj(handle,'Tag','edit1')`表示从句柄为 `handle` 的对象的下一级图形对象中查找 `Tag` 名称为 `edit1` 的图形对象。

（3）不同图形对象的 Callback 函数或不同窗口之间的参数传递

在用户界面程序设计中，我们经常会遇到一种情况，那就是需要在不同的图形对象的 Callback 函数或不同的窗口之间传递参数，这时候我们可以将需要传递的参数设置为全局变量。全局变量使用关键字 `global` 声明，并且在每个需要引用这个变量的函数种都要声明一次。

使用全局变量可以实现数据共享，但是也容易破坏程序的结构，同一个 GUI 界面中不同的图形对象的 Callback 函数间共享数据也可以利用 `handles` 全局架构变量实现，回调函数可以利用 `handles` 向其他回调函数传递变量，例如回调函数希望将变量 `X` 共享给其他函数，首先需要将 `X` 增加到 `handles` 中，这可以通过如下代码实现

```
handles.x_data = X;  
guidata(hObject, handles);
```

这里 `handles` 增加了一个域 `x_data`，其值为 `X`，这样其他函数就可以通过 `handles.x_data` 访问 `X`，从而实现数据共享。`guidata(hObject, handles)`用于保存所做的更改，否则其他函数不能访问 `handles.x_data`。

（4）在指定的坐标轴上绘制图形

当一个用户界面上存在多个坐标轴时，我们想要在其中指定的坐标轴上绘制图形，可以通过命令 `axes(handle)`将句柄为 `handle` 的坐标轴指定为当前坐标轴，然后编写绘制图形的代码。

6.对话框

MATLAB 提供了一些内置的对话框，可以借助对话框显示提示信息，或输入参数等，这里我们介绍一些最常用的对话框。

（1）菜单对话框

使用 menu 函数可以创建一个菜单对话框，使用方法如下

```
select = menu('显示的标题','选项 1','选项 2',……)
```

select 返回选项对应的索引值。下面为一个简单的范例

```
%Ex_A4_1.m
```

```
t=0:0.01:10;f=sin(t);
```

```
figure;
```

```
select=menu('请选择线型','实线','虚线');
```

```
switch select
```

```
    case 1
```

```
        plot(t,f);
```

```
    case 2
```

```
        plot(t,f,'--');
```

```
end
```

执行上述代码结果如图 9，选择“虚线”则返回 select=2，执行绘制虚线的代码，结果如图 10。

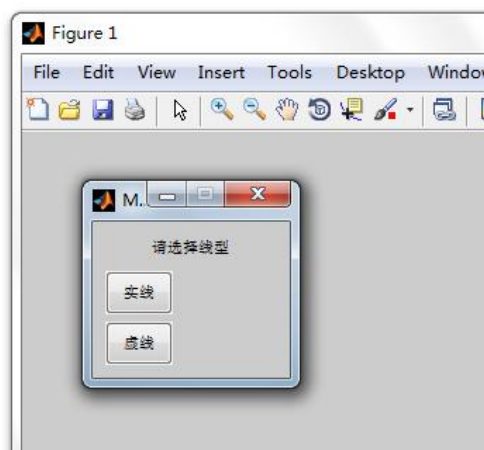


图 9 菜单对话框

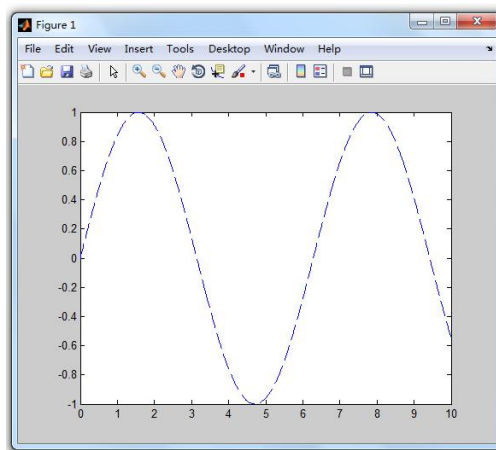


图 10 绘图结果

(2) 信息对话框

函数 msgbox 用于建立信息对话框，用法如下

```
h = msgbox('要显示的信息')
```

```
h = msgbox('要显示的信息','对话框标题','对话框图标')
```

对话框图标提供了一些常用选择：**none**（无图标）、**error**（错误提示图标）、**help**

（帮助提示图标）、`warn`（警告提示图标）等。错误提示、帮助提示和警告提示对话框还可以通过 `errordlg`、`helpdlg` 和 `warndlg` 函数实现，效果相同。

（3）问题对话框

函数 `questdlg` 用于建立问题对话框，用法如下

```
select = questdlg('标题','选项 1','选项 2',……,'默认选项')
```

我们来看一个示例程序

```
%Ex_A4_2.m
```

```
select = questdlg('请问你是男生还是女生？','请选择','男生','女生','女生')
```

```
switch select
```

```
    case '男生'
```

```
        msgbox('原来你是男生！','选择结果','help');
```

```
    case '女生'
```

```
        msgbox('哇！原来你是女生！','选择结果','help');
```

```
end
```

执行上述代码结果如图 11，选择“女生”则弹出信息对话框如图 12。



图 11 问题对话框



图 12 信息对话框

（4）输入对话框

函数 `inputdlg` 用于建立信息对话框，用法如下

```
answer = inputdlg({'提示语'},'标题','输入框行数',{'默认值'})
```

需要注意的是输入框返回的值为元胞数组类型，需要将元胞数组中的元素取出后方可进一步使用。我们来看一个范例

```
>> answer = inputdlg({'请输入 A','请输入 B'},'输入框范例',1,{'','2'})
```

显示如图 13 所示的输入框



图 13 输入框

在 A 的输入框中输入 5，然后单击 OK 按钮，命令窗口中显示结果

answer =

'5'

'2'

(5) 打开文件、保存文件对话框

MATLAB 提供了标准格式的打开文件对话框函数 `uigetfile` 和保存文件对话框函数 `uiputfile`

[文件名, 路径名] = `uigetfile` (文件类型, 对话框标题)

[文件名, 路径名] = `uiputfile` (文件类型, 对话框标题)

执行如下代码

```
>> [filename,filepath] = uigetfile('*.jpg;*.png','Open File');
```

显示如图 14 所示打开文件对话框，选择文件后单击打开按钮就会将文件路径和文件名分别以字符串的形式保存在变量 `filename` 和 `filepath` 中，然后我们可以根据文件路径和文件名采用文件操作的相关函数对文件进行操作。

执行如下代码

```
>> [filename,filepath] = uiputfile('*.m','Save File');
```

打开如图 15 所示保存文件对话框。

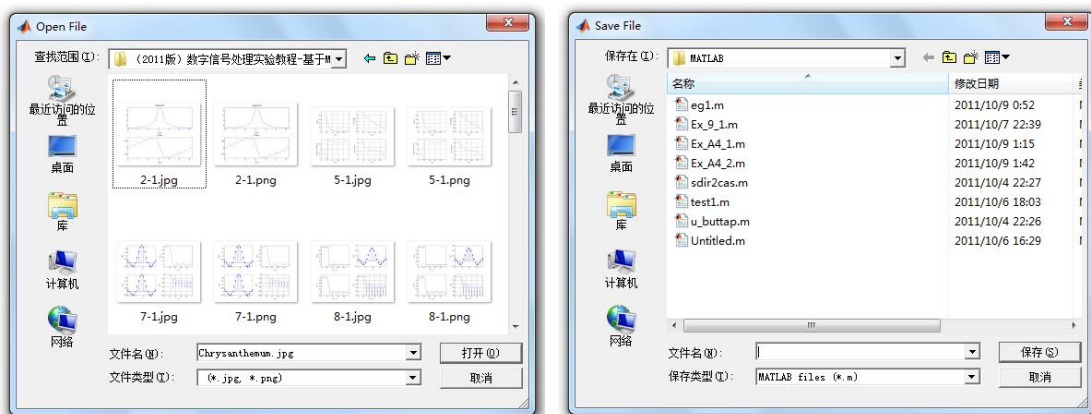


图 14 打开文件对话框

图 15 保存文件对话框

7.GUI 设计示例

我们设计一个 GUI 演示程序，允许用户从界面上输入连续时间系统系统函数的分子和分母多项式系数，根据输入的多项式系数绘制系统的频率响应波形。

首先，首先拖动需要的图形对象到界面布局区，并通过鼠标拖动和属性设置安排好布局。将静态文本框 `text1`、`text2`、`text3`、`text7` 的 `string` 属性分别设置为“分子多项式系数”、“分母多项式系数”、“系统函数为：H(s)=”和“提示：分子、分母多项式按降幂排列输入，元素间以空格隔开”，静态文本框 `text4`、`text5`、`text6` 的 `string` 属性设定为空，用于程序执行时显示系统函数表达式。将文本编辑框 `edit1`、`edit2` 的 `string` 属性设定为空，用于输入分子、分母多项式系数。将按钮 `pushbutton1` 和 `pushbutton2` 的 `string` 属性分别设置为“绘制频率特性曲线”和“退出程序”，将复选框 `checkbox1` 的 `string` 属性设置为“显示网格线”。设计界面如图 16 所示。

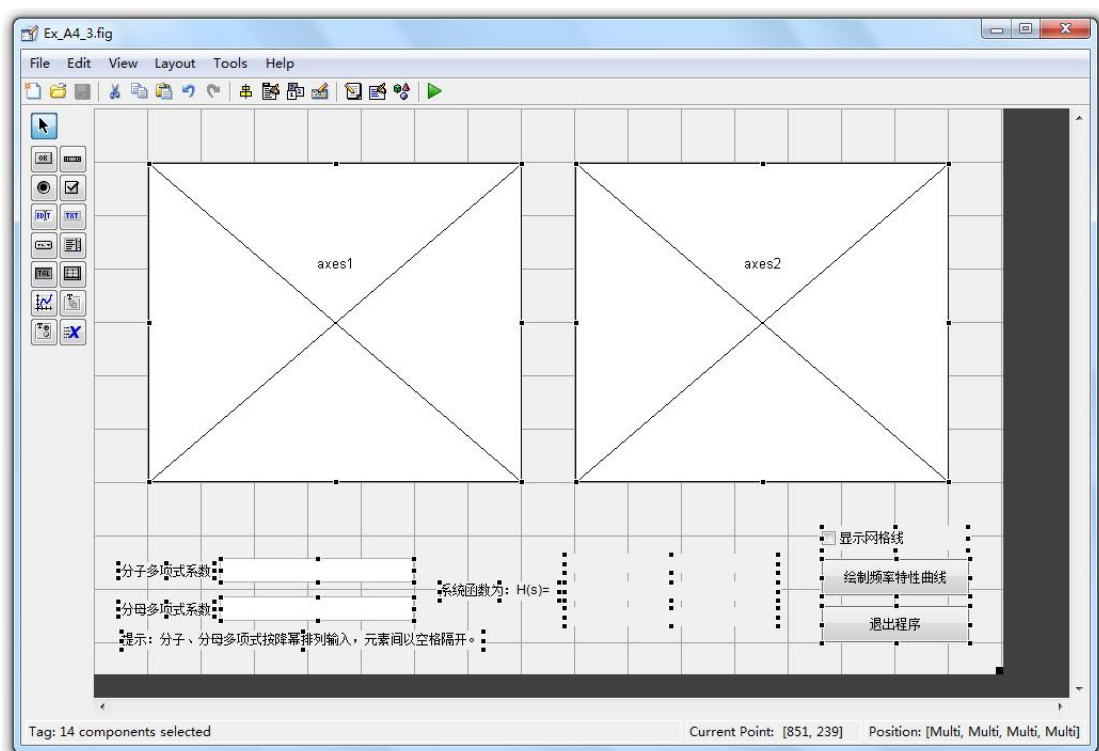


图 16 GUI 程序设计界面

接下来编写 `pushbutton1` 和 `pushbutton2` 的 `Callback` 函数，设定单击按钮时执行的操作，在设计界面中右键点击按钮对象，在右键菜单中选自“View Callback”→“Callback”即可开启 M 文件编辑器编辑 `Callback` 函数。分别编写这两个按钮对象的

Callback 函数如下：

```
% --- Executes on button press in pushbutton1.

function pushbutton1_Callback(hObject, eventdata, handles)

% hObject      handle to pushbutton1 (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      structure with handles and user data (see GUIDATA)

b = str2num(get(handles.edit1,'string'));    %读取从 edit1 输入的数值
a = str2num(get(handles.edit2,'string'));    %读取从 edit2 输入的数值
c = get(handles.checkbox1,'value');          %读取从 checkbox1 的状态

[H,w] = freqs(b,a);

axes(handles.axes1);          %选定 axes1 为当前坐标进行绘图
plot(w/pi,abs(H));
axis([0,2,min(abs(H)),max(abs(H))]);
xlabel('\omega/\pi');ylabel('Abs(H)');title('Magnitude');

axes(handles.axes2);          %选定 axes2 为当前坐标进行绘图
plot(w/pi,angle(H)/pi);
axis([0,2,min(angle(H)/pi),max(angle(H)/pi)]);
xlabel('\omega/\pi');ylabel('Angle(H)/\pi');title('Phase');

%根据 checkbox1 的状态确定是否显示坐标轴网格线
switch c
    case 0
        axes(handles.axes1);grid off;
        axes(handles.axes2);grid off;
    case 1
        axes(handles.axes1);grid on;
```

```

        axes(handles.axes2);grid on;
end

%显示系统函数表达式
num = char(poly2sym(b,'s'));    %分子多项式
den = char(poly2sym(a,'s'));    %分母多项式
num_s = size(num);
den_s = size(den);
s = max([num_s(2) den_s(2)]);
hl(1:s) = '-';
hl = char(hl);
set(handles.text4,'string',num);    %在 text4 中显示分子多项式
set(handles.text5,'string',hl);    %在 text5 中显示横线
set(handles.text6,'string',den);    %在 text6 中显示分母多项式

% --- Executes on button press in pushbutton2.
function pushbutton2_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
clear all;    %清空 Workspace
close;        %关闭窗口
然后为 checkbox1 编写 Callback 函数:
% --- Executes on button press in checkbox1.
function checkbox1_Callback(hObject, eventdata, handles)
% hObject    handle to checkbox1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

```

% Hint: get(hObject,'Value') returns toggle state of checkbox1

```
c = get(handles.checkbox1,'value');          %读取从 checkbox1 的状态
```

%根据 checkbox1 的状态确定是否显示坐标轴网格线

```
switch c
```

```
    case 0
```

```
        axes(handles.axes1);grid off;
```

```
        axes(handles.axes2);grid off;
```

```
    case 1
```

```
        axes(handles.axes1);grid on;
```

```
        axes(handles.axes2);grid on;
```

```
End
```

保存并运行程序，运行的用户界面程序如图 17 所示。在文本框中按规定的格式分别输入连续时间系统系统函数分子、分母多项式，单击“绘制频率特性曲线”按钮，即可在两个坐标轴中分别绘制系统频率响应的幅度和相位波形，复选框“显示网格线”可以控制坐标轴显示和关闭网格线，单击“退出程序”则退出 GUI 程序，演示程序运行界面如图 18 所示。

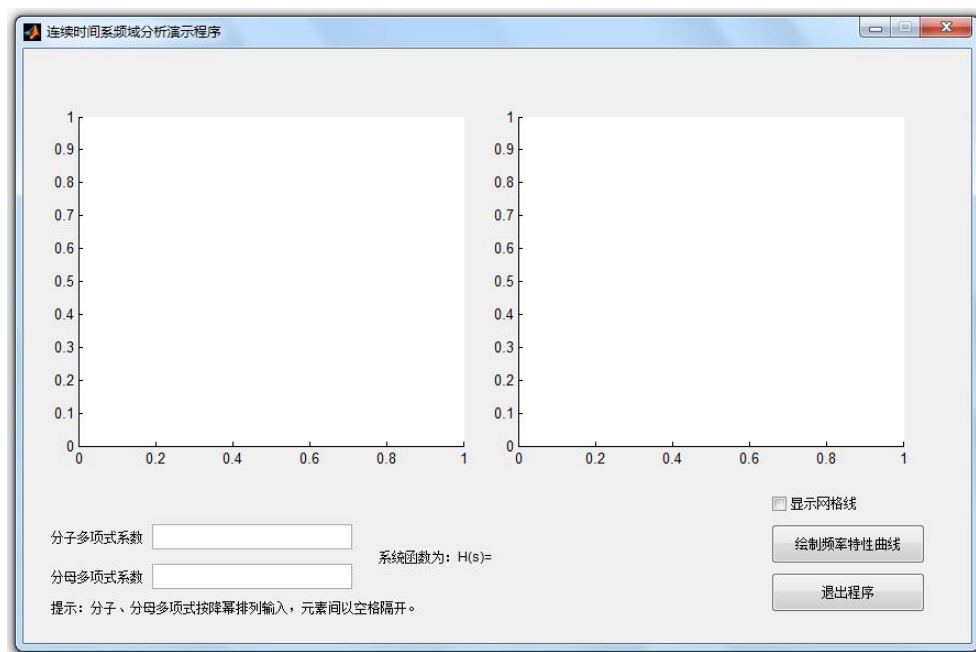


图 17 GUI 程序运行界面

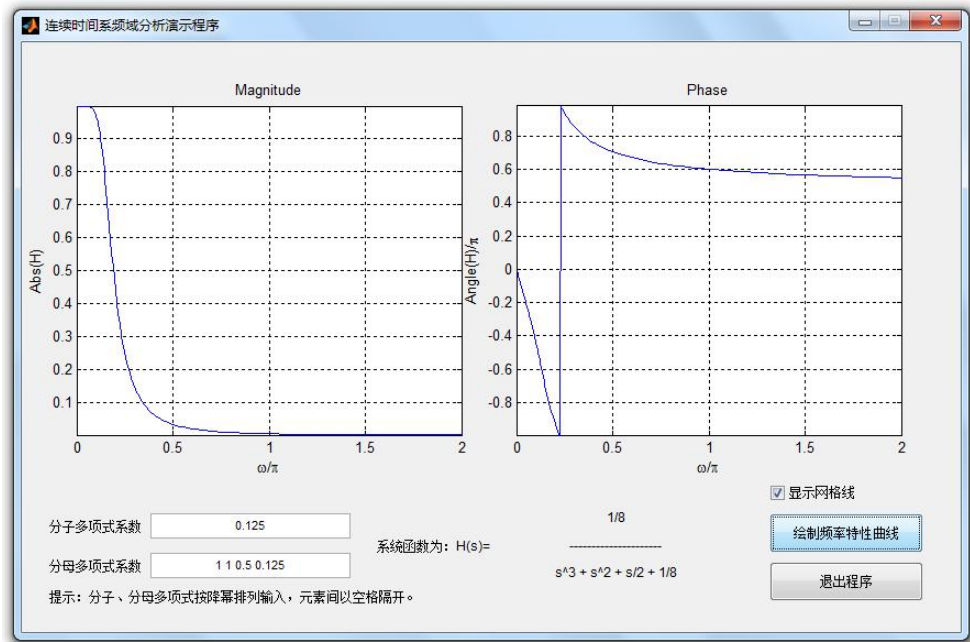


图 18 演示程序运行界面