3
图形句柄及其应用

Matlab 语言作为一种语句接近数学公式、编程便利、运算功能强大、绘图功能灵活多变、易于自学的开放式设计系统，其绘图动画功能虽不及 Flash 或 3ds max，但由于其强大的矩阵运算内嵌其中、与工程实际结合紧密等优点，使得工科绘图中多了一种选择。句柄图形（handle graphics）涉及 Matlab 图形系统中的底层（low-level）命令，可以完成许多命令无法实现的功能。本章结合 Matlab 中的句柄图形，首先简要介绍 Matlab 图形句柄的基本概念，然后介绍一些基本操作，最后以柴油机活塞在气缸中的运动动画制作为例说明图形句柄的应用。

本章主要内容：

- Matlab 句柄图形：图形对象的概念及体系结构。
- 图形对象的创建：各种类型的图形对象。
- 图形对象的属性：句柄的概念及其操作、对象属性的访问和设置。
- 默认属性。
- 其他功能介绍：菜单函数及属性编辑器。

3.1 Matlab 句柄图形

Matlab 中关于绘图的命令有很多，如二维绘图命令 plot、三维绘图命令 plot3、多面体绘图命令 cylinder 等，这些命令都处于 Matlab 图形系统中的高层（high-level）界面。使用这些命令虽然方便简单，但缺乏灵活性，因为我们并不清楚生成图形的细节。如果我们想用 plot 画出橘黄色的曲线，但 plot 命令中没有提供这种选项，而我们要用到的句柄图形（handle graphics）将涉及 Matlab 图形系统中的底层（low-level）命令，可完成前面命令无法实现的功能。

1. 图形对象

高层命令一般是对整个图形进行操作，但在句柄图形中，所有图形的操作都是针对图形对象而言的。所谓图形对象是指图形系统中最基本的单元，具体包括：根（计算机屏幕）、图形窗口、轴、线、块、面、像、文本、光线、用户界面控制框、用户界面菜单和用户界面隐含菜单 12 个对象。各图形对象之间的关系如图 3-1 所示。底层命令使用户可以对图形的一个或

几个对象进行独立操作，而不影响图形的其他部分，正是这种功能为绘图提供了极大的灵活性。图形对象如图 3-2 所示。

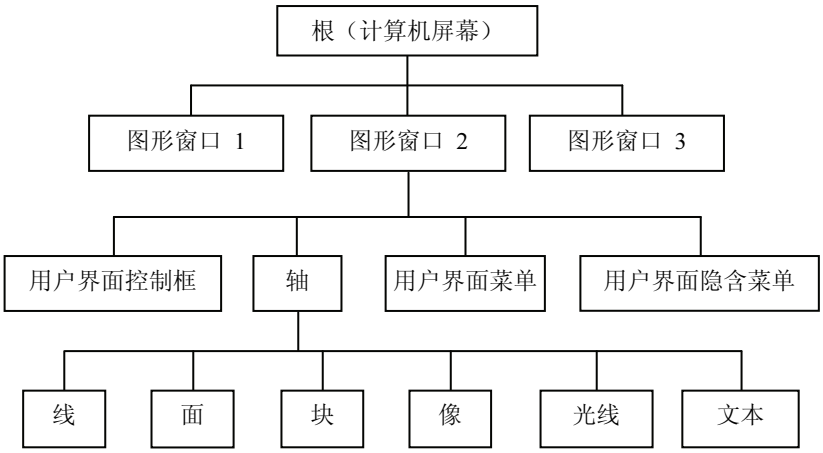

图 3-1　图形对象的结构

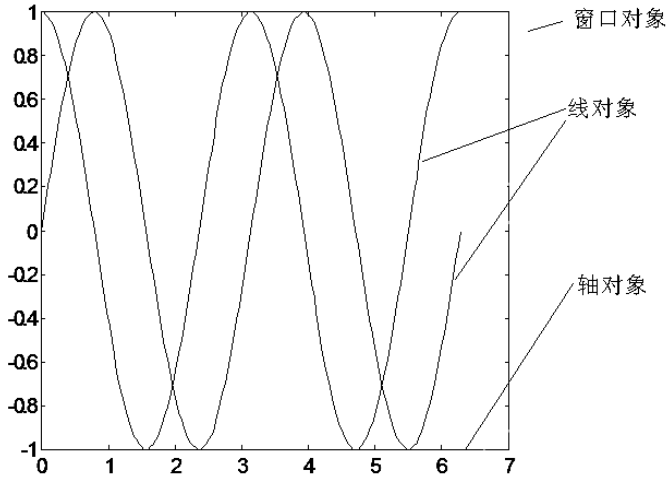

图 3-2　图形对象

2. 图形对象的句柄

每个具体的图形对象从它被创建时起就获得一个唯一的标志，这个标志就是该对象的句柄（handle），对每个对象进行操作都可以用它的句柄来识别。例如，根屏幕的句柄总是 0，图形窗口的句柄为一整数，其他对象的句柄是一些浮点数。但需要注意的是，这些浮点数可能很长，仅从屏幕上输出的数字来识别它们可能是不准确的，一定要把它们赋值于一个变量后引用，而不能根据屏幕显示的数据输入。

3. 图形对象的控制

指对已创建的对象进行的，如删除、保持、获取它们的句柄等操作。Matlab 为此提供了一系列控制函数来实现图形窗口的控制、轴控制以及其他图形对象的控制。

说明：

- 根：图形对象的根，对应于计算机屏幕，根只有一个，其他所有图形对象都是根的后代。
- 图形窗口：根的子代，窗口的数目不限，所有图形窗口都是根屏幕的子代，除根之外，其他对象都是窗的后代。
- 用户界面控制框：图形窗口的子代，创建用户界面控制对象，使得用户可以用鼠标在图形上作功能选择，并返回句柄。
- 用户界面菜单：图形窗口的子代，创建用户界面菜单对象。
- 轴：图形窗口的子代，创建轴对象，并返回句柄，是点面文本块像的父辈。
- 线：轴的子代，创建线对象。
- 面：轴的子代，创建面对象。
- 文本：轴的子代，创建文本对象。
- 块：轴的子代，创建块对象。
- 像：轴的子代，创建图像对象。

4. 图形对象的性质

所有图形对象都用属性来定义它们的特征，如 Root、Figure、Axes、Image 等，通过改变这些属性的取值可以修改图形对象的显示方式，我们研究的目的就在于此。每一种图形对象的属性包括属性名和与它们相关联的属性值。属性名是一个字符串，不同的属性值可能是不同类型的数，如实量、标量、整数、浮点、逻辑量、字符串等。在建立一个图形对象时，如不特别指定它的属性，就使用默认值。

5. 几个具体问题

（1）当前对象。

什么是当前对象？一个坐标中可能有很多个图形对象，究竟哪个是当前对象呢？一般说来，当前对象是指最后创建的对象。在计算机窗口中，两个窗口的叠放是很自然的，当前对象很容易判断；如果是两条交叉线就不同了，因为在两条线的交叉点上很难看清谁在上，谁在下。例如，用 line 产生两条线：H1=line([1 5],[1 5])、H2=line([2 4],[4 2])，很显然 H2 为当前对象。如果用鼠标单击 H1 线交叉点以外的地方，当前对象变为 H1，但如果单击交叉点，则当前对象始终为 H2。

（2）位置和单位。

许多图形对象的属性中都包含位置（Position）和单位（Units），位置属性都是一个四元素向量[left,bottom,width,height]，其中[left,bottom]对应于父对象左下角位置，而[width,height]是该对象的宽度和高度。

Units 是单位属性，可选值一般包括英寸、厘米、点、像素等。

（3）绘图。

高层绘图与底层绘图有什么区别？

- 高层绘图函数：是对整个图形进行操作的，图形每一部分的属性都是按默认方式设置的，充分体现了 Matlab 语言的实用性。
- 底层绘图函数：可以定制图形，对图形的每一部分进行控制，用户可以用来开发用户界面以及各专业的专用图形，充分体现了 Matlab 语言的开放性。

3.2 图形对象的创建

本节主要是在 3.1 节的基础上向用户详细介绍各种类型的图形对象。

1. 根对象

Matlab 中对象的最上层是唯一的一个对象，即 Root（根对象），在 3.1 节中提过，根对象是唯一的，其他所有对象均是根对象的各级子对象。根对象在 Matlab 启动时，由系统自动创建，直接对应计算机屏幕。虽然不是自行创建的，但用户可以对根对象的属性进行设置，从而影响图形的显示。

2. 图形对象窗口（Figure）

Matlab 根对象之下的第一级子对象是 Figure（图形窗口对象），图形窗口是用来在根对象（屏幕）上显示 Matlab 图形的独立窗口，图形窗口中可以包含数据图形和图形用户界面。用户可以在 Matlab 屏幕中创建任意多个图形窗口（除非受到计算机本身资源的限制），所有的图形窗口都是根对象的子对象，而所有其他的图形都是图形窗口的子对象。图形对象有两个基本部分：核心图形对象（Core graphics objects）和复合图形对象（Composite graphics objects），两者的具体解释如下：

- 核心图形对象：被高层绘图函数和合成对象用来创建图形对象（Plot objects）。
- 复合图形对象：由核心对象合成的并有机结合用来提供给用户的便捷绘图界面。

如果当前的 Matlab 还没有创建图形窗口对象，则调用任何一个绘图函数，如 plot、mesh 等，都可以让系统自动创建图形窗口；而如果当前屏幕已包含多个图形窗口，则总有一个窗口被定为当前窗口，作为所有绘图函数的输出窗口。

3. 用户界面对象（UI objects）

用户界面对象是图形窗口对象的一个子对象，用来创建用户界面的若干相关图形。以子对象 Uicontrol 为例，如果用户激活该对象，则系统执行相关的回调函数，生成多种类型的实例，如按钮、列表框、滑块等 Windows 对话框的基本选项。用户界面对象的其余子对象，读者可参考 Matlab 中相关的帮助文档。

4. 轴对象（Axes）

轴对象和用户界面对象是平行的兄弟关系。轴对象在图形窗口中定义一个特定的区域，并将自身所有子对象都限制在该区域内，其包含的 4 个子对象分别为：核心对象（Core objects）、图形对象（Plot objects）、组对象（Group objects）、注释对象（Annotation objects）。

（1）核心对象（Core objects）：包含基本的核心对象，如 line、text、axes、patch、rectangular 等，用于一般图形的绘制；较为特殊的核心对象，如 surface、images、light 等。虽然这些函数不会显示，但是将影响一些对象的属性设置，具体如表 3-1 所示。

表 3-1 各对象的功能

对象	功能
figure	创建图形窗口
uicontrol	图形界面控制
uimenu	创建用户界面菜单

续表

对象	功能
Axes	创建轴对象
line	创建线对象
patch	创建块对象
surface	创建面对象，是底层函数
light	创建灯光对象

下面给出部分函数的调用格式。

①figure：创建图形窗口。

调用格式：h=figure(n)

n 为窗口序号。

②uicontrol：图形界面控制。

调用格式：h=uicontrol('property',value)。

property/value 确定控制类型。

③uimenu：创建用户界面菜单。

调用格式：h=uimenu('property',value)

property/value 确定菜单形式。

④axes：创建轴对象。

调用格式：h=axes('property', {left,bottom,width,height})

left,bottom,width,height 定义轴对象的位置与大小。

【例 3-1】

```
axes('position',[0.1 0.1 0.5 0.2])
x=0:0.5:10;
y=x;
plot(x,y)
```

图形结果如图 3-3 所示。

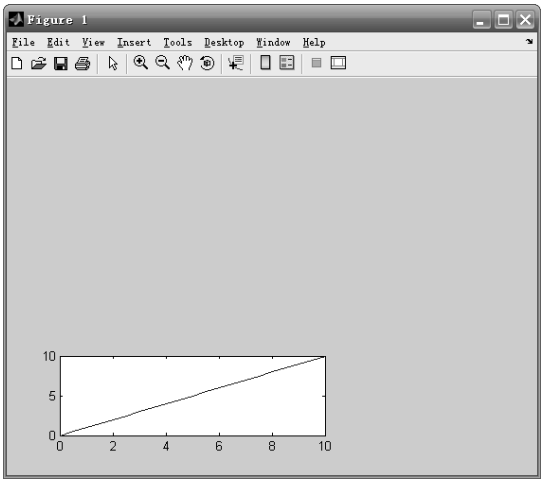

图 3-3　运行结果

axis 命令还可以定义轴的位置、宽度和高度。

如：axis([0 10 2 10])

注意二者的区别。

⑤line：创建线对象。

调用格式：h=line(x,y,z)

⑥patch：创建块对象。

调用格式：h=patch(x,y,z,c)

x、y、z 定义多边形，c 确定填充颜色。

⑦surface：创建面对象，是底层函数。

调用格式：h=surface(x,y,z,c)

x、y、z 为三维曲面坐标，c 为颜色矩阵，而 surf 是高级函数。

⑧light：灯光对象。

函数 light 创建一个灯光源，一个灯光源含 3 个因素：颜色、风格、位置。

调用格式：light('color',[1,1,1],'style',local or infinite,'position',[x,y,z])

local：x,y,z 表示光源位置；infinite：x,y,z 表示无穷远光通过该点射向原点。

【例 3-2】

```
subplot(2,2,1)
membrane                %这是一个库函数
light('color',[0.9 0.5 0.1],'position',[0,-2,1])
%风格省略为无穷远，光顺序通过(0,0,0)和(0,-2,1)
subplot(2,2,2)
membrane
light('color',[0.9 0.0 0.1],'style','local','position',[1,-1,1])
%风格为本地光，光源在(1,-1,1)位置
```

图形结果如图 3-4 所示。

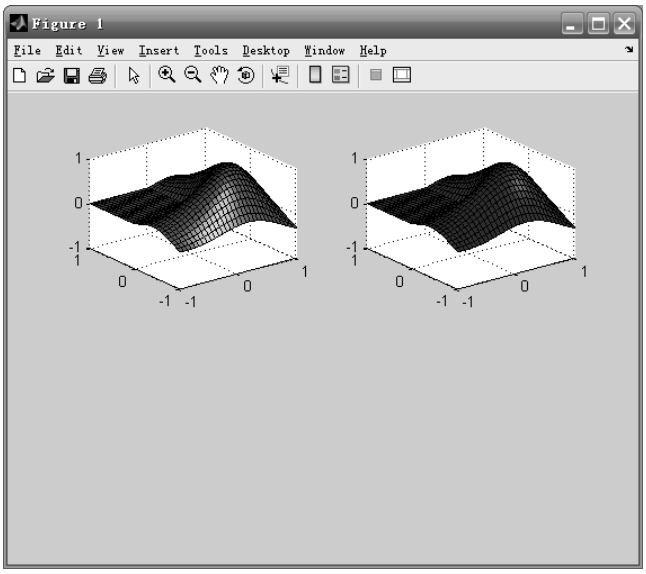

图 3-4　运行结果

⑨image：显示图像。

调用格式：h=image(x)

x 为图像矩阵。

⑩text：标注文字对象。

调用格式：h=text(x,y, 'string')

x、y 确定标注位置，string 为标注字符串。

如：h=text(0.1,0.2,'super star')

每个底层函数只能创建一个图形对象，并将它们置于适当的父辈对象中。

（2）图形对象（Plot object）：一些可以用高级绘图方式绘制图形的函数都可以返回对应的句柄值，从而创建图形对象。Matlab 中有些图形对象是由核心对象组成的，所以通过核心对象的属性可以控制这些图形对象的相关属性，其包含的绘制函数如表 3-2 所示。

表 3-2　图形对象包含的绘制函数

函数	功能
areaseries	绘制 area 图
barseries	绘制 bar 图
contourgroup	绘制 contour 图
errorbarseries	绘制 errorbar 图
lineseries	绘制曲线图
quivergroup	绘制 quiver 图
scattergroup	绘制 scatter 图
stairseries	绘制 stairs 图
stemseries	绘制 stem 图
surfaceplot	绘制 surf 图

（3）组对象（Group objects）：允许用户将轴对象的子对象设置为一个组，以便设置整个组内的对象属性。例如设置整个组为可见的（visible）或不可见的（invisible）。一旦选取了一个组对象，则其中的所有对象都将被选取。Matlab 中的组对象有两种：hggroup 和 hgtransform。当用户创建一个组对象并控制住对象的可见性或可选择性来作为一个独立对象时，使用前者；当组对象某些特性需要进行转换时，则使用后者。

（4）注释对象（Annotation objects）：在 Matlab 的注释对象中，line 和 Rectangle 与核心对象中的不同，读者要注意区分。用户可以通过图形绘制窗口的 Plot Edit 工具栏或菜单栏中的 Insert 选项来创建注释对象；另一种方式是通过 annotation 函数来创建。注释对象创建在隐藏的坐标轴中，既可以延伸宽和高在整个窗口中的显示，用户还可以通过正规化坐标（以左下角为原点(0,0)，右上角为(1,1)）的方式来定义注释对象在图形绘制窗口中的位置。

3.3　图形对象的属性

图形对象是由属性来描述的，可以通过修改属性来控制对象外观、行为等诸多特征。

用户不但可以查询当前任意对象的任意属性值，而且可以指定大多数属性的取值。在高

层绘图中对图形对象的描述一般是缺省的或由高层绘图函数自动设置的，因此对用户来说几乎是不透明的。

但在句柄绘图中上述图形对象都是用户经常使用的，所以要做到心中有数，用句柄设置图形对象的属性。

由于 Matlab 对象的默写属性属于公共属性，故这些属性的操作函数可使用所有 Matlab 中的对象，表 3-3 归纳了 Matlab 中的公共属性。

表 3-3　Matlab 中对象的公共属性

属性	描述
BusyAction	控制 Matlab 句柄回调函数的中断方式
ButtonDownFcn	单击按钮时执行回调函数
Children	该对象所有子对象的句柄
Parent	该对象的父对象
Clipping	能否剪切模式（仅对轴对象的子对象）
CreateFcn	同种类型的对象创建时执行回调函数
DeleteFcn	同种类型的对象被用户发出删除指令时执行回调函数
Handle Visibility	允许用户控制来自 Matlab 命令行和回调函数内部的对象句柄的可用性
HitTest	确定被鼠标单击选中的对象能否成为当前对象
interruptible	确定当前的回调函数是否可以被后续的回调函数中断
Selected	指出该对象是否被选中
SelectionHighlight	指定选中的对象是否可以可视化显示
Tag	用户指定的对象标签
Type	该对象的类型（Figure、Line、Text 等）
UserDate	用户希望与该对象关联的任意数据
Visible	指定该对象是否可见

下面将简要介绍句柄的基本概念，以及对象属性的获取与设置。

1. 句柄（handle）的基本概念

什么是句柄？句柄是图形对象的标识代码（唯一的身份），标识代码含有图形对象的各种必要的属性信息。

什么是句柄图形？句柄图形是利用底层绘图函数，通过对对象属性的设置与操作实现绘图。

句柄图形是一种面向对象的绘图系统，其中所有图形操作都是针对图形对象而言的。

句柄图形充分体现了面向对象的程序设计。

之前介绍的高层图形指令（如 plot）都是以句柄图形软件为基础写成的，也正是由于这个原因，句柄图形也被称为底层（Low-level）图形。

句柄图形的功能如下：

- 句柄图形可以随意改变 Matlab 生成图形的方式。

- 句柄图形允许定制图形的许多特性，无论是对图形做一点小改动，还是影响所有图形输出的整体改动。
- 句柄图形可以直接创建线、文本、网格、面、图形用户界面。

各图形对象的句柄数据格式：

根屏幕：0。

图形窗口：正整数，表示图形窗口序号。

其他对象：对应的双精度浮点数。

所有能创建图形对象的 Matlab 函数都可以给出所创建图形对象的句柄。

【例 3-3】创建 1 号窗口，返回句柄。

```
h=figure(1)
h=1    '返回值为窗口号数
h=figure('color',[1 0.1 0],'position',[0 0 200 100],'name','ww')
h=line(1:6,1:6)    '创建线对象的同时也建立了一个唯一的句柄
```

变量 h 是句柄值——符点数。

2. 当前对象属性的获得与设置

Matlab 中，有关句柄图形的一个极为重要的概念是当前性（Be Current）。例如，当前的窗口即为接收绘制函数输出的窗口；当前的坐标轴就是创建坐标轴子对象的命令输出目标坐标轴；当前的图形对象则为最后创建的图形对象。

用户可以直接把调用绘制函数的返回值存放在一个变量中，那么这个变量就是相应图形的句柄。

另外一种获取当前对象句柄的常用方法是调用 get 函数。

get：获得句柄图形对象的属性和返回某些对象的句柄值。

调用格式：get(gca,'属性')

返回当前坐标的单项属性值。

set：改变图形对象的属性。

专用函数：

gcf：当前窗口对象的句柄（Get Current Figure）。

gca：当前轴对象的句柄（Get Current Axes）。

get(gca)：返回当前坐标的所有属性值。

操作格式：

h=gcf：将当前窗口对象的句柄返回 h。

get(h)或 get(gcf)：查阅当前窗口对象的属性。

delete(gcf)：删除当前窗口的属性。

虽然 gcf 和 gca 提供了一个简单获取当前图形窗口对象和轴对象句柄的方法，但是却很少在 M 文件中使用，因为遵循一般设计的 M 文件不必根据用户行为来获取当前对象。

下面通过简单实例来说明两函数的使用方法。

【例 3-4】

```
x=1:10;
y=1:10;
h=line(x,y)
get(h)
```

```
get(gca,'children')          %轴的子代创建一个线对象并返回线对象的句柄值
h1=line([0:10],[0:10])
```

运行结果：

```
h1 =
   179.0144                  %h1 为句柄的代码值
```

结果如图 3-5 所示。

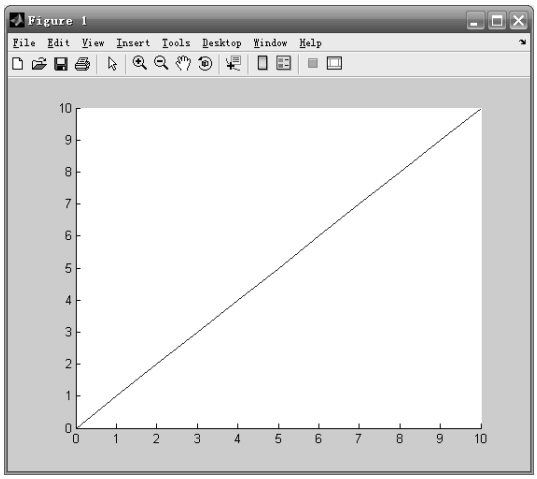

图 3-5　运行结果

```
%查阅线对象的属性名称和属性值
      get(h1)
      Color = [1 1 0]
      LineWidth = [0.5]
      MarkerSize = [6]
      Xdata = [    ]
      Ydata = [    ]
      Zdata = []
      Children = []
      Parent = [56.0001]
      Type = line
      UserData = []
%根据轴是线对象的父代，可查轴的句柄
      get(gca)
%可查色序
get(gca,'colororder')
```

运行结果：

```
ans =
         0         0    1.0000
         0    0.5000         0
    1.0000         0         0
         0    0.7500    0.7500
    0.7500         0    0.7500
    0.7500    0.7500         0
    0.2500    0.2500    0.2500
%设置线条和窗口的颜色
set(h1,'color',[1 0 0])              %如图 3-6 所示
pause(2)      %暂停 2s 观察图形的变化
```

```
set(h1,'color',[1 0.5 0])          %如图 3-7 所示
pause(2)
set(gcf,'color',[0.5 0.5 0.5])     %如图 3-8 所示
pause(2)
set(gcf,'color',[0.5 0.6 0.8])     %如图 3-9 所示
```

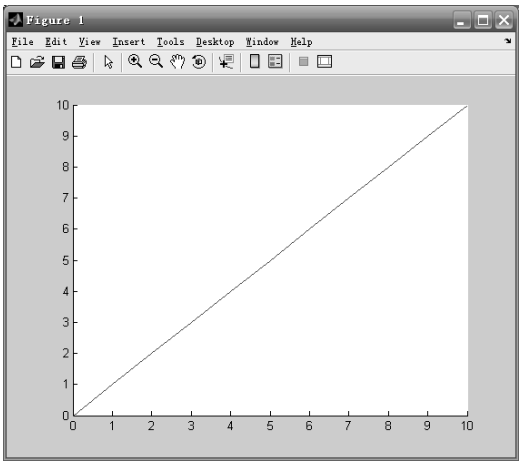

图 3-6　运行结果 1

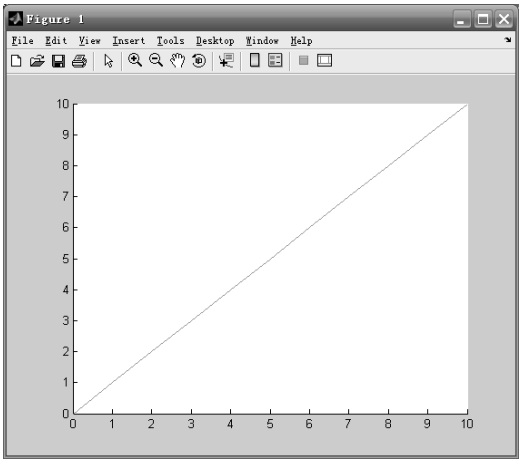

图 3-7　运行结果 2

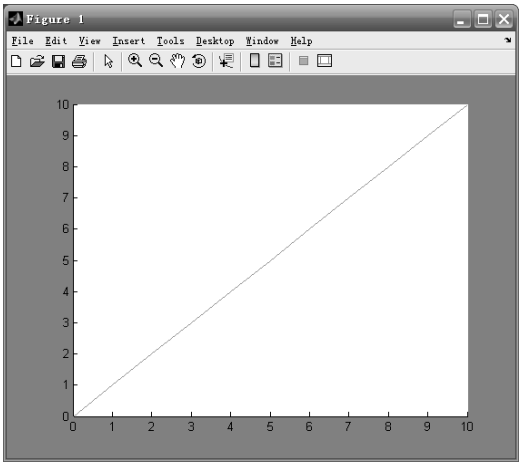

图 3-8　运行结果 3

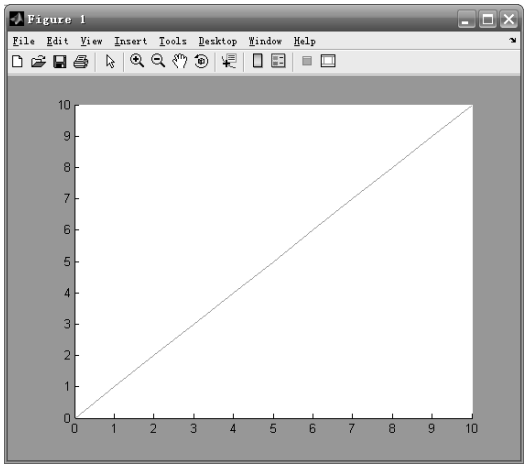

图 3-9　运行结果 4

3．对象的属性操作

控制一个图形对象是通过句柄实现的，具体是通过句柄操作函数 get、set 将某对象句柄属性进行设置与修改。

（1）对象属性的直接操作。

对象属性的直接操作是通过当前句柄来实现的，所以首先要获得当前句柄值以及对象的属性，然后再查询或修改。

```
get(h)
get(h, 'propertyname'）
set(h)
set(h, 'propertyname',value)
set(h, '属性名称', '新属性'）
```

```
'color', 'r'
'linestyle', ':'
'figurecolor','m'
```

（2）对象属性的继承操作。

对象属性的继承操作是通过父代对象设置默认对象属性来实现的。

父代句柄属性中设置默认值后，所有子代对象均可以继承该属性的默认值。

属性默认值的描述结构为：

Dfault+对象名称+对象属性

如：DefaultFigureColor：图形窗口的颜色。

DefaultAxesAspaceRatio：轴的视图比例。

DefaultLineLineWide：线的宽度。

DefaultLineColor：线的颜色。

默认值的获得与设置也是由 get、set 函数实现的。

get(0,'DefaultFigureColor')：获得图形窗口的默认值。

set(h,'DefaultLineColor','r')：设置线的颜色为红色。

例如，在图上添加文字注释，颜色为红色。

```
set(gca,'DefaultTextColor',[1 0 0])
gtext('正弦')
gtext('余弦')              %鼠标取点
```

在轴对象上设置字对象的颜色默认值为红色，继承该默认值在图上添加红色的文字注释。

【例 3-5】

```
%在轴对象（父代对象）上设置线的颜色默认值为红色
x=0:2*pi/180:2*pi;
y=sin(2*x);
set(gca,'DefaultLineColor',[1 0 0]);
h=line(x,y)
```

运行结果：

```
h =
    68.0001
```

图形如图 3-10 所示。

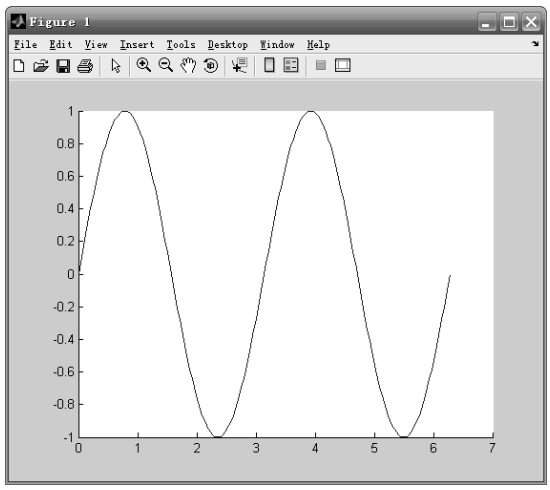

图 3-10　运行结果 1

```
set(h,'color','default')
%变成默认的红色
```

结果如图 3-11 所示。

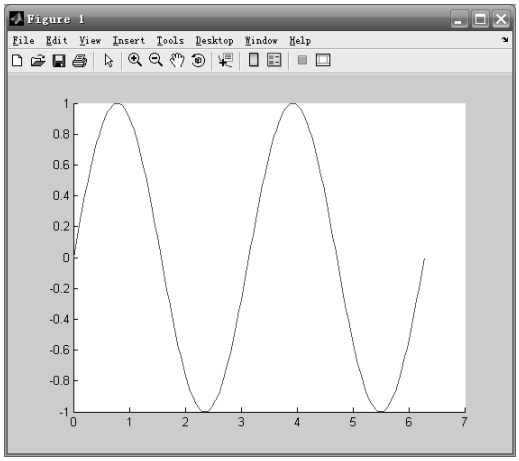

图 3-11　运行结果 2

【例 3-6】

```
x=0:2*pi/180:2*pi;
y=sin(2*x);
h=line(x,y)
set(0,'DefaultFigureColor',[0.5 0.5 0.5])
% 将所有新图形窗口的颜色由默认值黑色设置为适中的灰色
```

结果如图 3-12 所示。

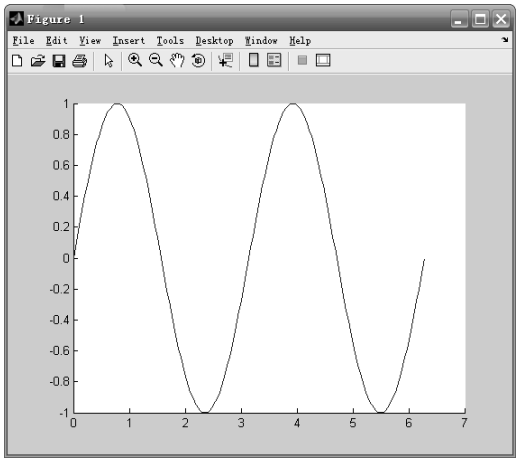

图 3-12　运行结果 1

```
set(h,'color','m','linewidth',2,'linestyle','*')
```

结果如图 3-13 所示。

```
set(0,'DefaultFigureColor','b')
h=line(x,y)
set(h,'color','r')
set(gca,'xcolor','w')
set(gca,'ycolor','w')
```

结果如图 3-14 所示。

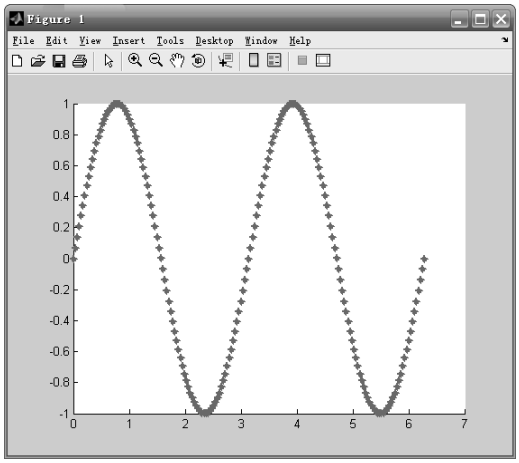

图 3-13　运行结果 2

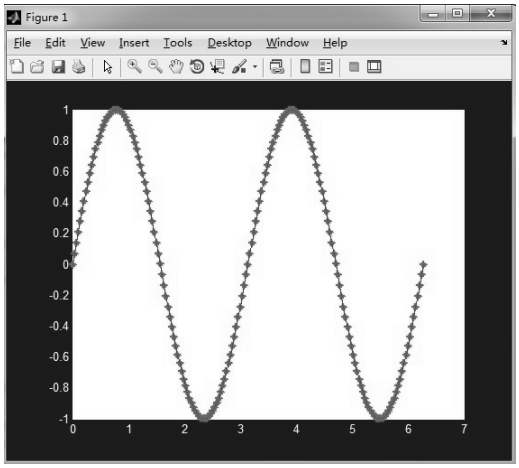

图 3-14　运行结果 3

3.4　默认属性

以上向读者简要介绍了 Matlab 对象属性的获取和设置，实际上，Matlab 中的所有对像属性都有系统内建的默认值，即出厂设置；当然，用户也可以自行定义任何一个 Matlab 对象的默认属性值。

1. 默认属性值的搜索

Matlab 对于默认值的搜索是从当前对象出发，沿着对象树型等级结构上行，直到发现用户自定义的默认值或出厂设置值。在定义对象默认值时，读者需要注意以下几点：

- 该对象的等级关系离 Root 越近，其作用范围就越广，如果用户在 Root 对象层面给线对象定义一个默认值，则 Matlab 将该值应用于所有的线对象；如果用户仅在轴对象层面给线对象进行了默认值的定义，则 Matlab 只将该值应用于当前的坐标系。
- 如果用户在不同层次上定义了同一个属性的不同属性值，则 Matlab 选择最下层的属性值作为最终的属性值。
- 自定义的属性值影响该属性创建之后的对象，而对于已经创建的图形对象不起作用。

2. 默认属性值的设置

指定 Matlab 对象的默认值，首先创建一个以 Default 开头的字符串，其格式为：中间为对象类型，其余部分为对象的属性名称，例如用户需要指定当前图形窗口对象层面上的线对象的线宽属性的默认值为 3，代码如下：

```
set(gcf,'DefaultLineLineWidth',3)
```

其中字符串“DefaultLineLineWidth”中第一个“Line”代表对象类型是线对象，“LineWidth”代表需要设置的对象属性名称是“LineWidth”，开头的“Default”是固定格式字符串。整个字符串“DefaultLineLineWidth”表示需要设定默认值的属性为线对象的线宽属性，句柄 gcf 设定了设置的默认值所在的对象层。如果用户输入如下代码，则为对图形窗口界面对象的色彩进行设置，使用了字符串“DefaultFigureColor”，此时句柄参数为 0，代表只

在根对象层指定图形窗口对象的色彩。

```
set(gcf, 'DefaultFigureColor', 'b')
```

另外，调用 get 函数还可以确定当前的默认值设置是在对象结构的哪一层面，例如如下代码的作用是返回当前图形窗口中的所有默认设置。

```
get(gcf, 'Default')
```

3. 对象属性的出厂设置

用户没有设定对象属性的默认值或不把属性值作为参数使用，Matlab 为其对象所有属性都设定了特定的值，一般把这个值称为“属性的出厂设置值（Factory-Defined Property Values）”，用户可以通过输入以下代码来获得对象属性的出厂设置值的完整列表：

```
a=get(0, 'factory')
```

代码中 get 函数返回一个构架数组，结果如下：

```
a =
                    factoryFigureAlphamap: [1x64 double]
                  factoryFigureBusyAction: 'queue'
               factoryFigureButtonDownFcn: ''
                    factoryFigureClipping: 'on'
            factoryFigureCloseRequestFcn: 'closereq'
                      factoryFigureColor: [0 0 0]
                   factoryFigureColormap: [64x3 double]
                  factoryFigureCreateFcn: ''
                  factoryFigureDeleteFcn: ''
                factoryFigureDockControls: 'on'
                    factoryFigureFileName: ''
          factoryFigureHandleVisibility: 'on'
                     factoryFigureHitTest: 'on'
               factoryFigureIntegerHandle: 'on'
               factoryFigureInterruptible: 'on'
              factoryFigureInvertHardcopy: 'on'
        ……%篇幅限制，部分属性值省略
                   factoryRootInterruptible: 'on'
                factoryRootRecursionLimit: 2.1475e+009
           factoryRootScreenPixelsPerInch: 96
            factoryRootSelectionHighlight: 'on'
             factoryRootShowHiddenHandles: 'off'
                          factoryRootTag: ''
                     factoryRootUserData: []
                     factoryRootVisible: 'on'
```

所有出厂设置值都包含在该构架数组的相应字段中，例如：

```
factoryRootVisible: 'on'
```

以上显示的代码表明根对象（Root）的 Visible（是否可见）属性的出厂设置值为 on，即根对象出厂设置其“是否可见”的属性默认值为“可见”。

另外，用户还可以单独获取个别属性的出厂设置值，get 函数的调用格式为：

```
get(0, 'Factory<ObjectType><PropertyName>')
```

例如，用户希望查询获取 Figure 类型对象的 Color 属性，则只需输入如下代码：

```
>>get(0, 'FactoryFigureColor')
  ans=
       0     0     0
```

其中调用格式中的<ObjectType>为 Figure，<PropertyName>为 Color。

3.5 其他功能介绍

1. 菜单函数 menu

调用格式：K= MENU(HEADER, ITEM1, ITEM2, ...)

如：K = menu('请选择','plot','mesh','surf')

结果如图 3-15 所示。

图 3-15 运行结果

2. 属性编辑器

Propedit(h)：打开属性编辑器。

【例 3-7】

```
x=0:2*pi/180:2*pi;
y=sin(2*x);
h=line(x,y)
set(0,'DefaultFigureColor',[0.5 0.5 0.5])
Propedit(h)
```

结果如图 3-16 所示。

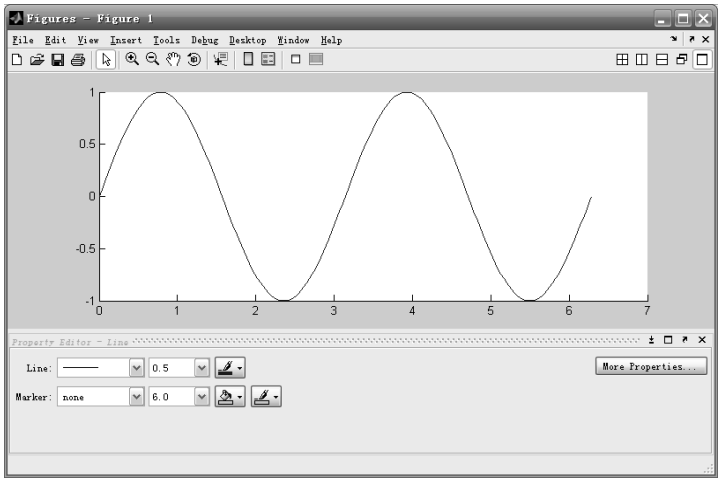

图 3-16 运行结果

下面以例 3-8 来说明 Matlab 句柄图形在图形处理中的应用。

【例 3-8】柴油机活塞曲柄连杆机构运动，活塞 P 在固定的气缸 C 中运动，如图 3-17 所示。图中气缸为固定部分，曲柄旋转时是圆周运动，绕固定支点旋转；将活动的图形对象分成

5 个：活塞 P、活塞轴 S1、连杆 L、曲柄 r、连杆和曲柄铰链 S2。动画中只要使这 5 个图形对象活动即可。建立如图 3-7 所示的坐标系，坐标原点在曲柄运动中心，决定 5 个活动对象位置的关键几何量是 S1 和 S2 的高度，令 S2 在圆周（曲柄运动轨迹）上旋转的变量是 dt，则有：

$$h = [l^2 - r^2\cos^2(dt)]^2 + r\sin(dt)$$

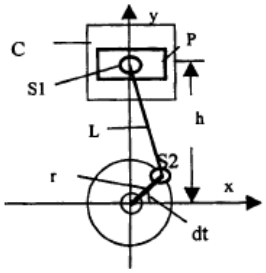

图 3-17　曲柄连杆机构

在用底层绘图函数制作动画时，要寻找一个变量，这里是圆角 dt，然后把活动图形对象的位置 x、y 表示为 dt 参数，这就是程序中的 xp、yp，xs1、ys1，xs2、ys2，xa、ya，xa1、ya1；接着用 set 命令通过句柄把这些数据矩阵赋给各对象，再用 drawnow 命令绘制出来。因为动画的实质是不断擦去旧图形，所以对擦除的方式和速度都存在要求，在程序中选择了最快的擦除模式。

具体的运行界面如图 3-18 所示。

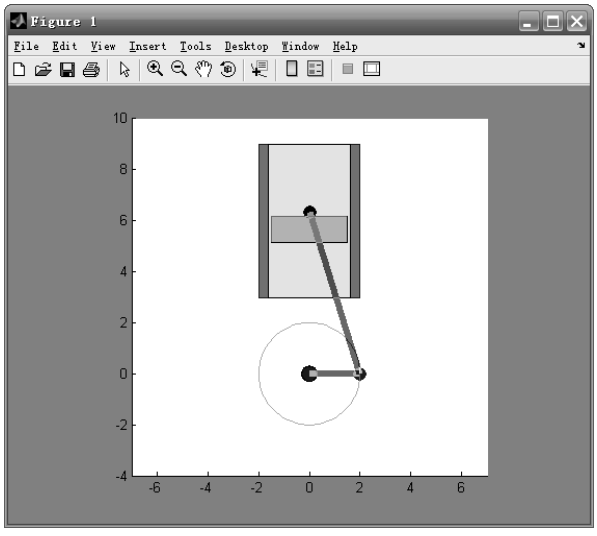

图 3-18　运行界面

其实现的源代码如下：

```
clf;
Hf=figure(1);
Ha=axes('PlotBoxAspectRatio',[1,1,1],'Visible','on');
%draw cylinder
x=[-2,-1.6,-1.6,-2]';y=[3,3,9,9]';
Hp1=patch(x,y,[1,0.2,0.2],'EdgeColor','b');
```

```
x=[1.6,2,2,1.6]';
Hp2=patch(x,y,[1,0.2,0.2],'EdgeColor','b');
x=[-1.6,1.6,1.6,-1.6]';
Hp3=patch(x,y,'y');
%draw the track of crank
t=[0:0.01:1]*2*pi;
H1=line(2*sin(t),2*cos(t),'LineWidth',1,'Color',[0.5,0.7,1]);
%draw whell shaft
Hp4=patch(0.3*sin(t),0.3*cos(t),'b');
%set axis,make it equal and unvisible
axis([-7,7,-4 10]);
%create piston
r=2;l=6;dt=0;tt=0;
h=r*sin(tt+dt)+sqrt(l*l+r*r*cos(tt+dt)^2);
x0=[-1.5,1.5,1.5,-1.5]';
y0=[h-0.5,h-0.5,h+0.5,h+0.5]';
piston=patch(x0,y0,'g','EdgeColor','b','EraseMode','xor');
%create shafts
haft1=line(0,h,'Color','b','Marker','.',...
'MarkerSize',30,'EraseMode','xor');
shaft2=line(r*cos(tt+dt),r*sin(tt+dt),'Color','b',...
'Marker','.','MarkerSize',30,'EraseMode','xor');
%create arm
arm=line([2*cos(tt+dt),0],[2*sin(tt+dt),h],'LineWidth',5,...
'Color',[0.4 0.3 1],'EraseMode','xor');
arm1=line([0,2*cos(tt+dt)],[0,2*sin(tt+dt)],'LineWidth',5,...
'Color',[0.4 0.3 1],'EraseMode','xor');
%animation
for i=0:600
dt=pi/10;dsin=sin(i*dt);dcos=cos(i*dt);
h=r*dsin+sqrt(l*l-r*r*dcos*dcos);
xp=x0;yp=[h-0.5,h-0.5,h+0.5,h+0.5]';
xs1=0;ys1=h;
xs2=r*dcos;ys2=r*dsin;
xa=[r*dcos,0];ya=[r*dsin,h];
a1=[0,r*dcos];ya1=[0,r*dsin];
set(piston,'XData',xp,'YData',yp);
set(shaft1,'XData',xs1,'YData',ys1);
set(shaft2,'XData',xs2,'YData',ys2);
set(arm,'XData',xa,'YData',ya);
set(arm1,'XData',xa1,'YData',ya1);
drawnow;pause(0.0005);
end;
```