

Einführung in Matlab

Einheit 6

Jochen Schulz

Georg-August Universität Göttingen 

14. September 2009

- 1 Grafik-Handle
- 2 Grapical User Interface (GUI)

1 Grafik-Handle

2 Grapical User Interface (GUI)

- Jedes Grafik-Objekt ist eindeutig bestimmt durch seine Eigenschaften (engl. *properties*).
- Die Eigenschaften der Objekte sind in sogenannten *handles* gespeichert. Sie liegen dort als `double` (Gleitkommazahl) vor.
- Mit Hilfe dieser Handle können die Eigenschaften existierender Grafik-Objekte geändert werden.

Hierachische Struktur von Grafik-Objekten

Level 1	Root	Das Wurzel-Objekt bezieht sich auf den Computerschirm. Es wird automatisch erzeugt und es gibt nur eins. Alle anderen Objekte stammen von diesem ab.
Level 2	Figure	Diese Objekte sind durch die einzelnen Grafik-Fenster gegeben. Alle Figuren sind Kinder von Root.
Level 3	Axes, Uicontrol, Uimenu, Uicontextmenu	Die Objekte Ui... sind benutzerdefinierte Grafik-Interfaces (werden hier nicht betrachtet). Die Axes Objekte definieren eine Region im Grafikfenster und ordnen ihre Kinder in dieser Region an.
Level 4	Image, Line, Text, Surface,...	Die Objekte bestimmen das Aussehen der Grafik-Fenster. Sie sind Kinder der Axes Objekte.

Umgang mit dem Grafik-Handle

- Konstruktion einer Grafik

```
>> x=0:0.2:2*pi; plot(x,sin(x))
```

- Abfragen und Bedeutung der Handles aller Objekte

```
>> h=findobj
h =
         0
    1.0000
  100.0015
    3.0016
```

```
>> get(h,'type')
ans =
    'root'
    'figure'
    'axes'
    'line'
```

Umgang mit dem Grafik-Handle

- Momentane Einstellung des 'Axes'-Objekts

```
>> set(h(3))  
    ActivePositionProperty: [position | {outerposition}]  
    ALim  
    ALimMode: [ {auto} | manual ]  
    AmbientLightColor  
    ...
```

- Eigenschaften des 'Line'-Objekts

```
>> set(h(4))  
    DisplayName: {}  
    Color: {}  
    LineStyle: {5x1 cell}  
    LineWidth: {}  
    ...
```

Umgang mit dem Grafik-Handle

- Ändern des Markers:

```
set(h(4), 'Marker', 's', 'MarkerSize', 4)
```

- Ändern der Einheiten auf der x-Achse

```
set(h(3), 'XTick', [0 pi/2 pi 2*pi])  
set(h(3), 'XtickLabel', '0|pi/2|pi|2pi')
```

- gca, gcf und gco sind die Handle für die aktuelle *Axes*, die aktuelle *Figure* und das aktuelle *Objekt* des 4. Levels.

```
set(gcf, 'Name', 'Tolle Abb.')  
set(gca, 'FontSize', 15)
```

```
l = legend('sin(x)');  
set(l, 'FontSize', 20);
```


- Die Grafikobjekte sind hierarchisch angeordnet. Sie haben also Kinder und Väter.
- Informationen zu den zugeordneten Kindern

```
a=get(1, 'Children'), get(a, 'type')
```

- Information zum Vater

```
d=get(1, 'Parent'), get(d, 'type')
```

- Ändern der Kindeigenschaften

```
set(a(3), 'Color', [1 0 0])
```

Beispiel

```
%-----  
% current_figure.m  
%  
% Aendert die Ueberschriften der Figures so ab,  
% das jeweils das aktuelle Fenster die Ueberschrift  
% 'aktuell' hat und die anderen 'nicht aktuell'  
%-----  
  
% Handle aller Figures  
a = get(0, 'children')  
  
% Beschrifte alle Figures als 'nicht aktuell'  
for i = 1:length(a)  
    set(a(i), 'name', 'nicht aktuell')  
end  
  
% Ueberschreibe den Namen des aktuellen Fensters  
set(gcf, 'name', 'aktuell')
```

- Löschen von Objekten: `delete(handle)`
- Kopieren von Objekten: `copyobj(handle,new_parent)`. Hängt das Objekt mit Handle `handle` an einen anderen Vater `new_parent` an.
- Finden von Objekten mit bestimmten Eigenschaften:
`findobj(Eigenschaft, Spezifikation)`.

- Ansehen der Defaultwerte

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

- Ändern der Defaultwerte

```
set(0,'DefaultLineLineWidth',3)  
set(0,'DefaultFigureColor','g')  
set(0,'DefaultAxesFontSize',20)
```

- Einstellungen gelten immer auch für alle Kinder und Kindes-Kinder.

Defaulteinstellungen II

- Löschen der Defaulteinstellung

```
set(0, 'DefaultAxesFontSize', 'remove')
```

- Die Defaulteinstellungen können in der Datei startup.m (Linux/Unix) bzw. matlabroot\toolbox\local (Windows) abgelegt werden. Sie werden so beim Start von MATLAB automatisch eingeladen.
- Unter Linux muss die Datei durch den Suchpfad path erreichbar sein (oder einfach dort hinlegen: ~/.matlab/startup.m)

Ein Beispiel

```
% This example shows how the properties of a
% graphic can be modified
close all;

% Generate Grid
x = linspace(-2,2,30);
[X,Y] = meshgrid(x,x);

% Function Values
Z = exp(-X.^2-Y.^2).*sin(pi*X.*Y);

% Plot graphic
h = surf(X,Y,Z);

k = get(h,'Parent'); % Handle for the graphics object
Angles = get(k,'View');

for l = 1:360
    set(k,'View',Angles+l);
    pause(1/60)
end
```

1 Grafik-Handle

2 Grapical User Interface (GUI)

- Das Graphical User Interface(GUI) ermöglicht das Steuern von Programmen mit Hilfe der grafischen Oberfläche.
- Hierdurch können Programme von Usern ohne MATLAB-Kenntnisse genutzt werden. Weiterhin kann dies die Benutzung auch für den Implementierer von numerischen Algorithmen hilfreich sein.
- Im Wesentlichen werden GUIs durch Grafik-Objekte der Typen `uimenu`, `uicontextmenu` und `uicontrols` gesteuert. Sie sind auf der gleichen Hierarchie-Ebene wie `axes`-Objekte.
- Es existiert eine grafische Oberfläche zur Erstellung von GUIs. Aufruf `guide`

Vordefinierte GUIs für Dialogboxen

- `helpdlg`: Hilfebox
- `msgbox`: Eine beliebige Nachricht
- `warndlg`: Anzeige einer Warnung
- `inputdlg`: Abfragen einer Größe
- `questdlg`: Frage

Beispiel:

```
h1 = warndlg('NameFenster','Nachricht')
h2 = errordlg('NameFenster','Nachricht')
ans = questdlg('NameFenster','Nachricht')
ant = inputdlg({'Frage 1','Frage 2','Frage3'},...
    'NameFenster',[1 2 3], {'defAnt1','defAnt2','defAnt3'})
```

- **uicontrols**: Interaktive grafische Objekte, die Aktionen steuern oder bestimmte Optionen setzen.
- **uimenu**: Benutzerdefinierte Menüführung in einer `figure` (wird nicht behandelt).
- **uicontextmenu**: Ein Pop-up Menü, das erscheint, wenn der Benutzer mit der rechten Maustaste ein grafisches Objekt anklickt (wird nicht behandelt).

Arten von UiControls

'checkbox'	Wahl von Zuständen an/aus
'edit'	Editierbare Texteingabe
'frame'	graf. Gruppierung von Kontrollen
'popup'	Auswahl aus Liste bei Anklicken
'listbox'	Auswahl aus einer skrollbaren Liste
'pushbutton'	Starten eines Events
'radio'	Auswahl einer Option
'toggle'	Wahl des Zustandes: an/aus
'slider'	Auswahl aus Wertebereich
'text'	Anzeige von Text in einer Box

Eigenschaften von UiControls

Style	Art des UiControls
CallBack	Durch Anklicken des Users auszuführende Aktion
Position	Lage des Uicontrol-Objekts in der figure Eingabe: [links unten Breite Höhe] Einheiten werden durch die Units-Eigenschaft gesteuert.
String	Textdarstellung. Bei mehreren Optionen durch Eingabe <code>string={'opt1'; 'opt2'}</code> oder <code>string='opt1 opt2'</code>
Units	Einheit zur Bestimmung der Lages des UiControls, Werte: <code>pixels</code> (Default), <code>centimeters</code> , <code>normalized</code> (Werte in $[0, 1]$)
Tag	String zum Auffinden des Objekts durch <code>findobj</code>

Erzeugen von UiControls

UiControl-Objekte können durch `uicontrol` erzeugt werden. Aufruf:

```
handle = uicontrol('eig1','wert1',...  
                  'eign','wertn')
```

Beispiel:

```
>>u = uicontrol('style','listbox',...  
'string','Option1|Option2|Option3',...  
'units','centimeters','position',[0 0 3 3])
```

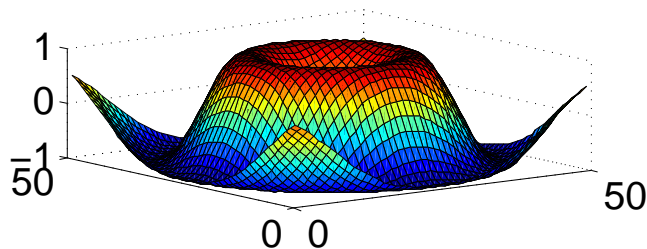
- Sobald ein GUI-Objekt aktiviert wird, führt es MATLAB Code aus. Dies wird *Callback* genannt.
- *Callback* ist eine Eigenschaft von UIControl.
- Ein GUI-Objekt wird in der Regel durch einen Mausklick oder ähnliches aktiviert.
- Callback kann eine Funktion oder ein String, der durch eval ausführbar ist, sein.
- Syntax der Callback-Funktion

```
function varargout = objectTag_Callback(h,ev_data,...  
                                         handles, varargin)
```

- objectTag ist der Name, der im Tag des Objects gespeichert wird.
- h ist der Handle des Objekts, dass die Callback-Funktion aufruft.
- ev_data wird nicht benötigt (ist reserviert).
- handles Struktur aller Objekte in der GUI
- varargin ist eine Liste von Argumenten, die an die Funktion übergeben wird.

- Callback-Funktionen werden immer im Haupt-Workspace gestartet. Übergabe von Variablen schwierig.
- Mit Hilfe der `figure`-Eigenschaft `UserData` können Daten an das `figure`-Objekt gehängt werden.
- Der User kann grafische Objekte per Hand schließen. Dies kann zu falschen Aufrufen führen.
- Man achte darauf, dass man die richtigen Objekte anspricht. Tipp: Eigenschaft `Tag`.

Beispiel einer GUI

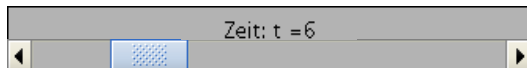


Darstellung

☒ Surf

☐ Mesh

☐ Contour



Aktualisieren

bild_funktion.m

```
function han = bild_funktion()
%----- Hauptprogram
x = linspace(-1,1,50);
y = linspace(-1,1,50);
t = 0:1:30;
A = erzDaten(x,y,t);
han = erzGUI(A);
end
%----- Grafische Oberflaeche erzeugen
function han = erzGUI(A);
delete(findobj('tag','figGUI'));
fig = figure('name','Beispiel GUI','UserData',A,'tag','figGUI');
han.pushbutton = uicontrol(fig,'Parent',fig,'Style',...
    'pushbutton','units','normalized','position',...
    [0.8 0.2 0.15 0.15],'String','Aktualisieren',...
    'Callback','darstGrafik');

han.grafikachse = axes('Position',[0.1 0.5 0.6 0.3],'tag',...
    'axesGUI');
han.grafik = surf(A(:,:,1));
```

```
han.frame1 = uicontrol(fig,'style','frame', 'units',...  
    'normalized','position',[0.1 0.2 0.6 0.1]);  
han.slider = uicontrol(fig,'style','slider','sliderstep',...  
    [0.2 0.2], 'min',0, 'max',30, 'units','normalized',...  
    'position', [0.1 0.2 0.6 0.05], 'tag','slider',...  
    'Callback','darstGrafik');  
han.text1 = uicontrol(fig,'style','text', 'tag','text1',...  
    'units','normalized','position', [0.3 0.25 0.1 0.03],...  
    'String','Zeit t = 0');
```

bild_funktion.m

```
han.frame2=uibbuttongroup('units','normalized','tag','radio',...
    'position',[0.8 0.5 0.15 0.3]);
han.text2=uicontrol(fig,'style','text','parent',han.frame2,...
    'units','normalized','position',[0.1 0.6 0.8 0.3],...
    'String','Darstellung');
han.radio1=uicontrol(fig,'style','radio','parent',han.frame2,...
    'tag','r1','units','normalized',...
    'position',[0.1 0.45 0.8 0.15],'String','Surf');
han.radio2=uicontrol(fig,'style','radio','parent',han.frame2,...
    'tag','r2','units','normalized',...
    'position',[0.1 0.25 0.8 0.15],'String','Mesh');
han.radio3=uicontrol(fig,'style','radio','parent',han.frame2,...
    'tag','r3','units','normalized',...
    'position',[0.1 0.05 0.8 0.15],'String','Contour');
end
function A = erzDaten(x,y,t)
[X,Y,T] = meshgrid(x,y,t);
A = cos(pi*T.^0.5.*exp(-X.^2-Y.^2));
end
```

```
function darstGrafik()

A = get(findobj('tag','figGUI'),'UserData');

t = round(1+get(findobj('tag','slider'),'Value'));
set(findobj('tag','text1'),'string', ...
    strcat('Zeit: t = ',num2str(t-1)) );
selection = findobj('tag','radio');
switch get(get(selection,'SelectedObject'),'tag')
    case 'r1'
        surf(A(:,:,t));
    case 'r2'
        mesh(A(:,:,t));
    case 'r3'
        contour(A(:,:,t));
    otherwise
        error('Keines oder zuviele entspr. GUIs geoeffnet');
end;
```