Dr. Frank Schöpfer

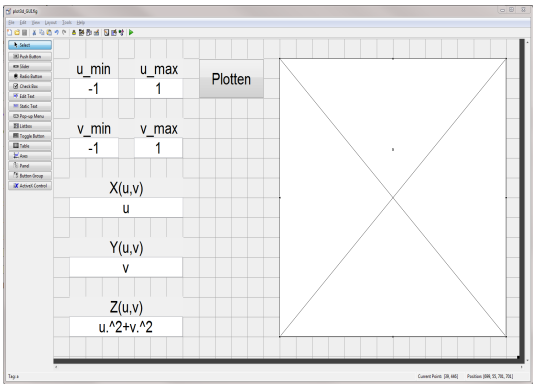
Einführung in Matlab

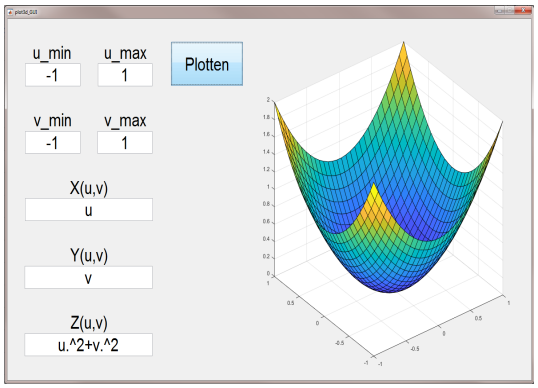
Übung 11

Aufgabe 1: Erzeugen Sie eine GUI namens `plot3d_GUI`, bei welchem der Anwender eine **Fläche im $\mathbb{R}^3$ plotten** kann. Bearbeiten Sie die Teilaufgaben nacheinander und testen Sie alles geeignet.

(a) Die GUI soll **folgende Komponenten und Eigenschaften** besitzen:

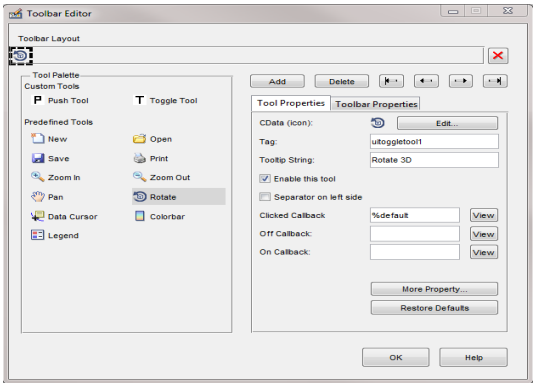
- **Edit Text-Boxen** für die Eingabe der Grenzen u_{min}, u_{max}, v_{min}, v_{max} der Parameter u und v, sowie der Koordinaten-Funktionen $x(u,v)$, $y(u,v)$ und $z(u,v)$.
  Im **Property-Inspector** jeweils **FontSize=30**, entsprechenden **Tag** und passenden **String** eingeben (z.B. wie im Bild unten für Paraboloid).
- Durch **Static Text-Boxen** soll jeweils verdeutlicht werden, was in die Eingabeboxen gehört.
- Bei Drücken des **Push Button** mit **String=Plotten** wird die Fläche in die vorhandene **Axes** geplotted. **Ergänzen Sie dabei die Callback-Funktion des Push Button** so, dass
  - mit **meshgrid** ein 30 mal 30 Gitter für die Parameter u, v in den Grenzen u_{min}, u_{max} und v_{min}, v_{max} erzeugt wird,
  - jeweils **aus dem String der Edit Text-Box ein Functionhandle** für die Koordinatenfunktionen erzeugt wird, welches von u und v abhängt,
  - und damit die **Fläche geplotted** wird (z.B. mit `surf(...)` um analog zum plot-Befehl ein surface-Objekt in eine Axes mit Voreinstellungen wie Kamarawinkel etc. zu plotten).

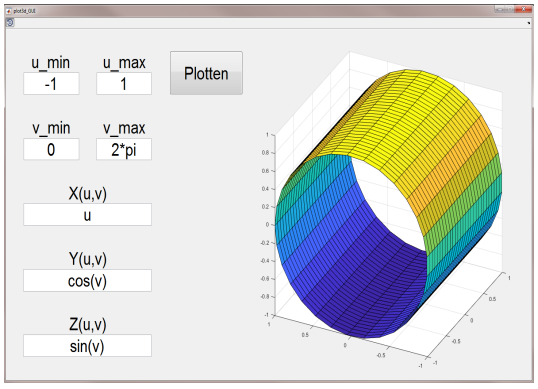




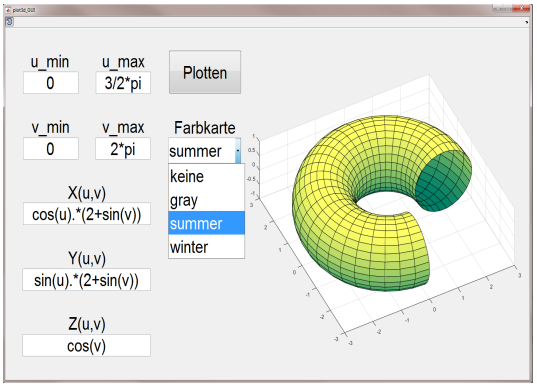
(b) Ergänzen Sie die **Opening-Function** der GUI so, dass **zu Beginn schon der Plot zu den vorgegebenen Strings erscheint**.

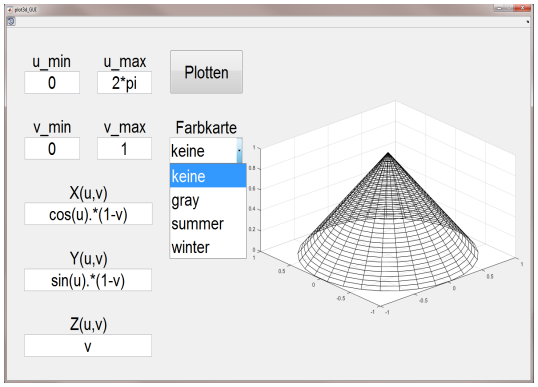
(c) Erweitern Sie die GUI um eine **Toolbar** (Tools -> Toolbar Editor) und wählen Sie z.B. das **Rotate-Icon**.



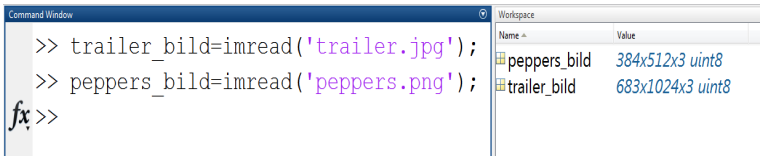


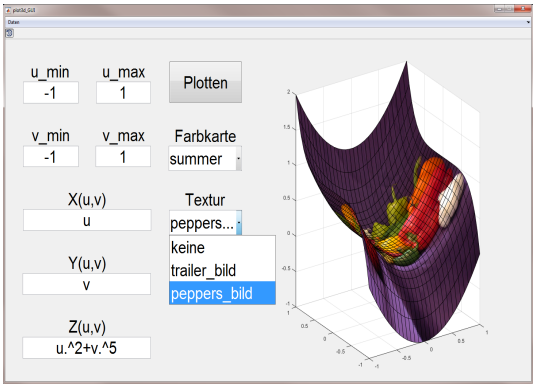
(d) Erweitern Sie die GUI um ein **Popup-Menu** bei dem je nach Auswahl die entsprechende **Farbkarte** `gray`, `summer`, `winter` verwendet wird **oder** bei `keine` **nur Gitterlinien**.
Hinweis: Bei Verwendung von Farbkarten muss für das surface-Objekt s gelten s.FaceColor='flat' und s.CData=s.ZData, und bei nur Gitterlinien s.FaceColor='none'.





(e) Fügen Sie ein weiteres **Popup-Menu** hinzu, bei dem je nach Auswahl die entsprechende **Textur** verwendet wird.
Hinweis: Der String `t_name` der Textur soll dabei der Name einer gültigen Bild-Array-Variablen im Workspace sein. Die Variable lässt sich dann im Function-File der GUI zuweisen durch den Befehl `t=evalin('base',t_name)` (siehe auch Hilfe zu `evalin`).





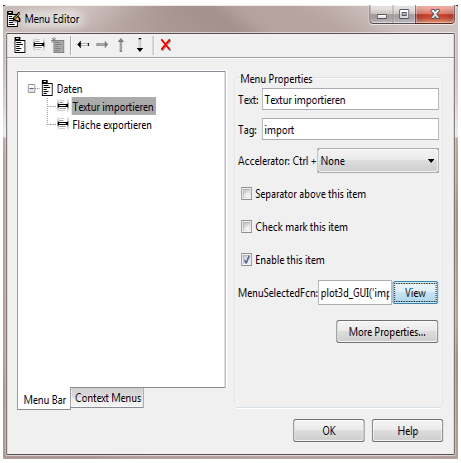
(f) Erweitern Sie die GUI um ein **Menu** (Tools -> Menu Editor): Mit **New Menu** `Daten` und **2 New Menu Items** `Textur importieren` und `Fläche exportieren`.
Ergänzen Sie dazu die **Callback-Funktionen** (Anzeigen der Callback-Funktionen durch View-Button im Menu Editor drücken) **folgendermaßen**, und **machen Sie sich den Code klar** (siehe auch die Hilfe zu den einzelnen Befehlen `evalin,who,assignin,listdlg,inputdlg`):

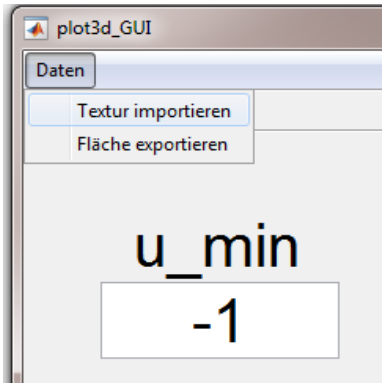
- ```
  function import_Callback(hObject, eventdata, handles)
  ...
  vars=evalin('base','who'); % Cell-Array mit Variablennamen als Strings
  if numel(vars)>0
   [ind,ok]=listdlg('PromptString','Variablen auswählen','ListString',vars);
   if ok
      n=numel(handles.t.String);
      for k=1:numel(ind)
          handles.t.String{n+k}=vars{ind(k)};
      end
    end
  else
      errordlg('Im Workspace sind keine Variablen!')
  end
  ```
  ```

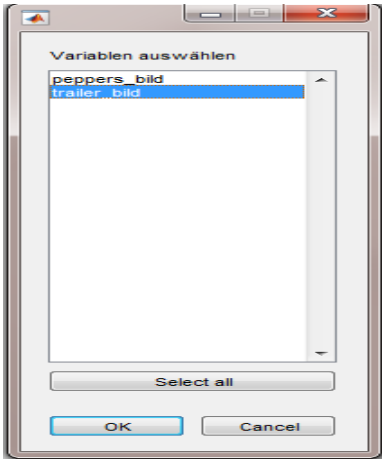
- ```
  function export_Callback(hObject, eventdata, handles)
  ...
  var_name=inputdlg('Variablenname:');
  if ~isempty(var_name)
      s=handles.a.Children; % surface-Objekt
      assignin('base',var_name{1},s);
  end
  ```

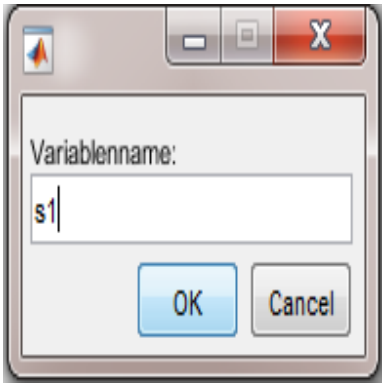
- Um **sicher zu gehen, dass bei String im Popup-Menu auch ein Cell-Array** ist, ergänzen Sie auch noch die **Create-Function des Popup-Menu** zu

  ```
  function t_CreateFcn(hObject, eventdata, handles)
  ...
  hObject.String={'keine'}; % so sicher ein Cell-Array
  ```








  ```