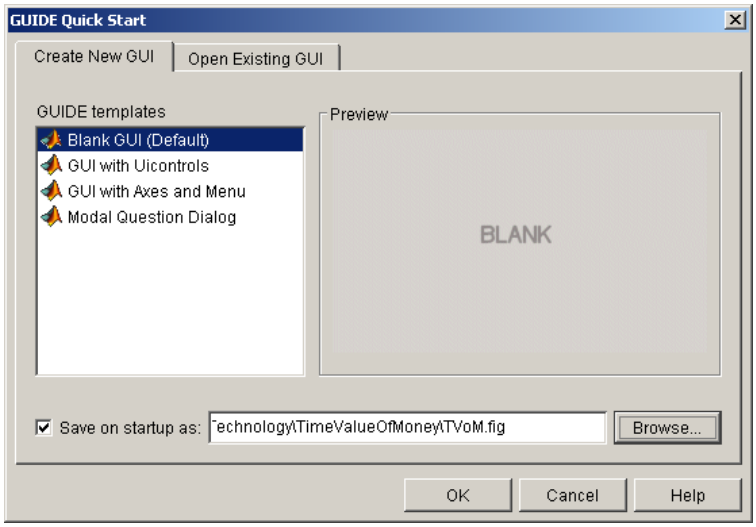
Guidelines on Developing Electrical Engineering Tutorials using MATLAB GUIs

This document consists of two sections. The first section, *An introduction to MATLAB's GUI Development Environment*, briefly explain the basic idea behind the MATLAB GUI Development Environment to an audience that has never used GUIDE, but is familiar with MATLAB's programming language.

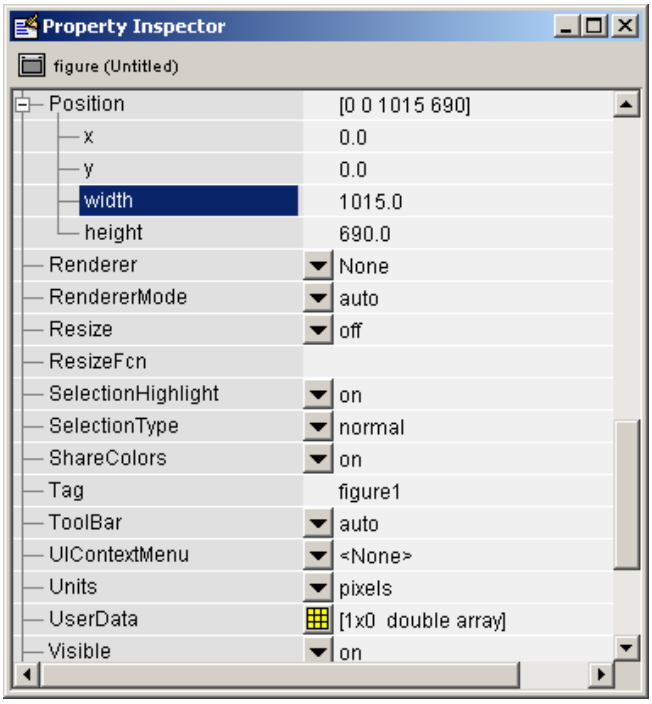
The second section, *Proposal for a future GUI*, presents and explains a potential MATLAB GUI tutorial that will be beneficial to the teaching process in Electrical Engineering, if developed.

An introduction to Matlab's GUI Development Environment

1. Open Matlab.
2. Type 'guide' at the command prompt. The window below will appear.

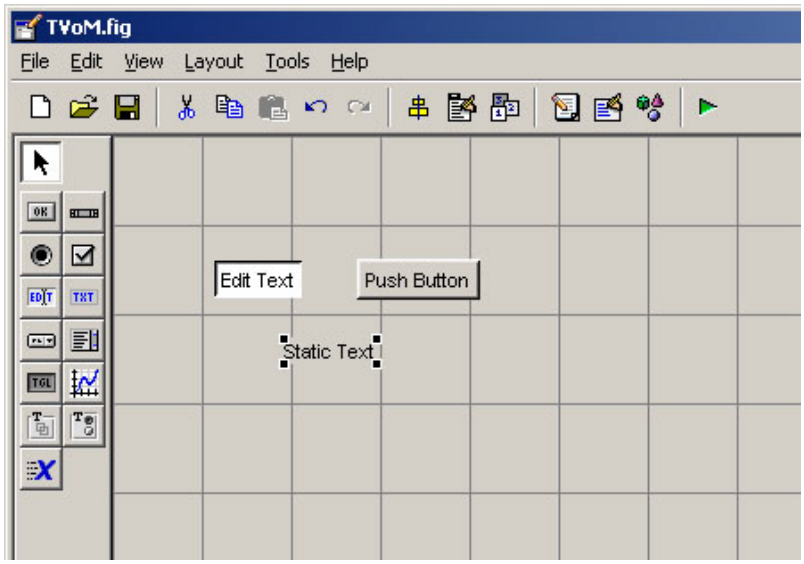


3. Select "Blank GUI (Default)" and type a new name for your project in the "Save on startup as:" textbox. Click on "OK"
4. Right-click on the resulting workspace, and select "Property Inspector". Change the "Units" property to "pixels", and thereafter the "Position" property to x=0, y=0, width=1015, height=690, thereby setting the resolution of the resulting program window to work on any screen of 1024x720 or larger.

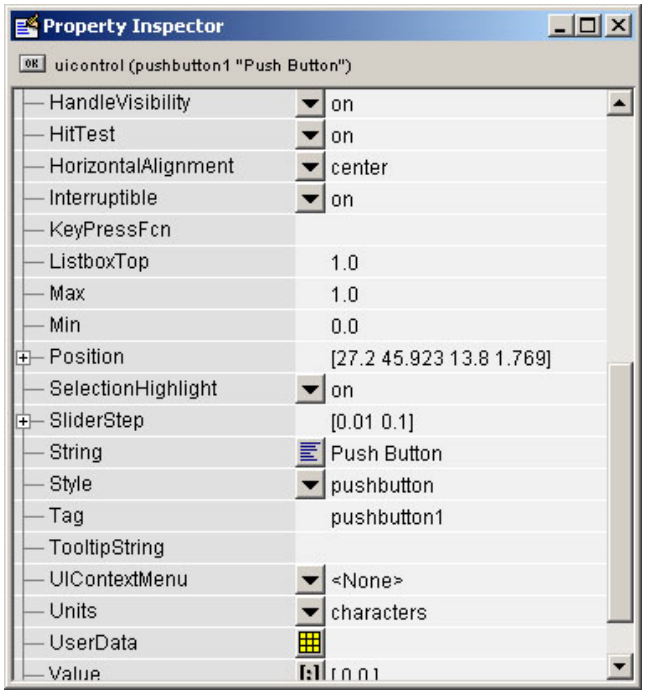


5. You have now created the basic background which you will next populate with a variety of text boxes, graph axes etc. The next few steps will demonstrate how the graphical objects interact with the Matlab code.

6. Drag and drop three objects from the objects toolbar (of the left) onto the background created in step 4: an “Edit Text”, “Static Text” and “Push Button” object, as shown below.



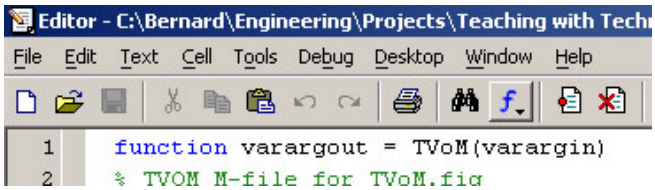
7. Every created object has a “Tag” property in the property inspector window (right-click on the chosen object, and select “Property Inspector”), with which it is referenced in the code. For example: the “Push Button” object has been allocated the tag “pushbutton1” as shown below.



8. Once the user executes / runs the program and actions an object, e.g. by clicking on the “Push Button” object, the code associated with the object’s tag is executed. The code can be edited as follows:

    a. Open the Matlab M-file editor by selecting “View” and “M-file editor” from the drop-down menubar of the GUIDE window.

    b. Click on the “show functions” button (marked with an blue “f”) on the top toolbar, as indicted below, and select the “pushbutton1 Callback” function.



    c. The default code that will be executed when the “Push Button” is clicked will appear, as shown below (an empty function, so nothing will happen when you click on the object during execution).

```
% --- 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)
```

    d. Edit this function to read as shown below. When the “Push Button” object is pressed, the string content of the “Edit Text” object is read, converted into a number and multiplied by 5, and written to the “Static Text” object as a string. Make sure that the tag used to address each object is correct, by verifying the tag name of each object in the property inspector of the GUIDE window.

```
% --- 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)

    %read 'Edit Text' object value into variable called num_temp
    num_temp=str2double(get(handles.edit1,'String'));
    %display num_temp time 5 in the 'Static Text' object
    set(handles.text1,'String',num2str(num_temp*5));
```

    e. You can now run / execute you program, by selecting “Debug” and “Run” from the Editor window, or “Tools” and “Run” from the GUIDE window. Any errors will be shown in the Matlab command line.

9. In the same way as illustrated in the above example toggle buttons, radio buttons etc. can be used. Please refer to the comprehensive Matlab Help function for more details.
10. Some tips:
    - If you which to define global variables in your program (i.e. variables that are visible within all functions), place the prefix “handles.” in front of the variable, e.g. handles.num_temp for the example in the previous section. Also ensure that your save the global variables each time a function completes, by adding the line “guidata(hObject, handles)” to the end of the particular function where the variable was updated.
    - When you change the “Tag” property of an object to a more descriptive name, make sure that the “Callback” and “CreatFcn” properties have also been updated to the new name. This should happen automatically, but for some reason often does not.
    - The Debug functionality of the Matlab Editor can be very helpful – refer to the Matlab Help for more details.

Proposal for a future GUI

Third year electrical engineering students often struggle to understand the effects of different 3-phase transformer wiring configurations on the resulting voltage / current waveforms.

This proposed software and associated tutorial worksheet will offer the student a simulation environment in which he / she can change the wiring configurations of the 3-phase transformers graphically, and immediately see the resulting change in the form of phasor diagrams and voltage / current waveforms.

Referring to the diagram on the next page, the student starts by selecting whether to select a standard designation from a list (e.g. DYN5), or to connect the windings manually.

If selecting from a list, the transformer diagram's wiring is updated automatically to reflect the student's choice. If selecting to connect manually, the student enters the connections using the diagram to the right of the transformer diagram, with the transformer diagram updating as the student enters his/her preference.

At the same time as updating the transformer diagram, the software will also draw the primary, secondary and combined phasor diagrams, and show the primary and secondary waveforms of each phase graphically. This will allow the student to grow in understanding the impacts of different transformer connections on the phasor diagrams and waveform shapes.

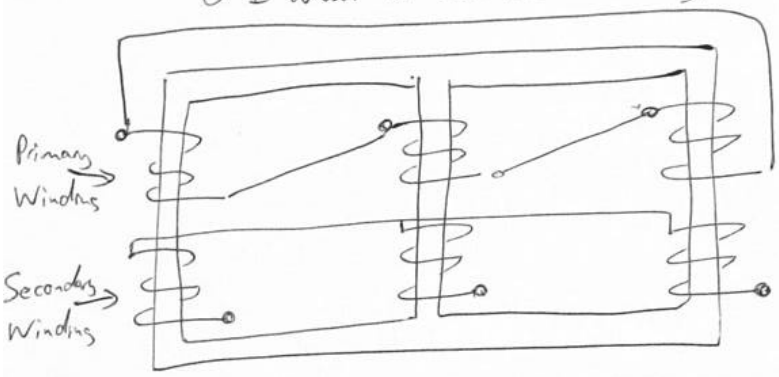
The above described proposed software, with its associated worksheet, is only a first draft of a potential tutorial. Shaping this proposal into a finished project will necessarily improve on this first draft.

3 ϕ Transformer Designations

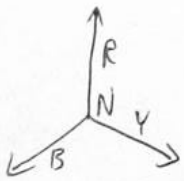
⊙ I want to select designation from list

○ I want to connect windings manually

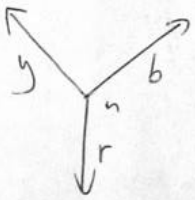
P1+	S1+
P2+	S2+
P3+	S3+
P1-Out	S1-Out
P2-Out	S2-Out
P3-Out	S3-Out



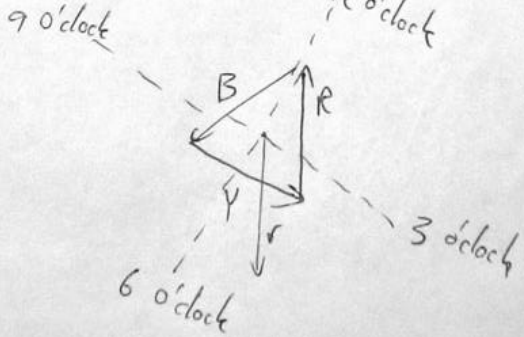
Phasors - Primary



Phasors - Secondary

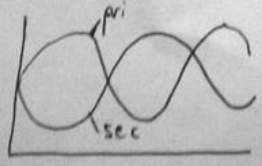


Phasors - combined



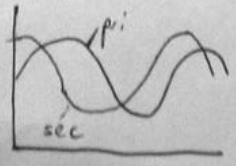
Waveforms ϕ 1

ϕ shift 120°



ϕ 2

ϕ shift 60°



ϕ 3

ϕ shift 30°