IMAGINE User's Manual

IMAGe visualization, analysis and evaluation engINE

 Contact: christian.wuerslin@med.uni-tuebingen.de

04/24/13

Table of Contents

Command Line Interface

This is a repetition of the header of the IMAGINE.m file.

`IMAGINE` starts the IMAGINE user interface without initial data

`IMAGINE(DATA)` Starts the IMAGINE user interface with one (DATA is 3D) or multiple panels (DATA is 4D).

`IMAGINE(DATA, TITLE)` Same functionality as above, however supplying a caption for the data.

`IMAGINE(DATA1, DATA2, ...)` Starts the IMAGINE user interface with multiple panels, where each input can be either a 3D- or 4D-array. No captions can be supplied with this call.

`IMAGINE(DATA1, TITLE1, DATA2, TITLE2, ...)` Starts the IMAGINE user interface with multiple panels, where each DATA input can be either a 3D- or 4D-array. Furthermore, captions for each input data array must be supplied (data and titles must be supplied in pairs).

`HA = IMAGINE(...)` Starts the IMAGINE user interface and returns the handles to the N axes created during startup. Thus, further plots can be overlaid to the imagine axes such as quiver plots. You can use the same comfortable zooming and windowing functions that imagine offers in conjunction with a variety of MATLAB visualization features. N depends on the number of input arguments and their dimension (just like in during normal startup. NOTE: Using this syntax, changing the amount of image panels during run-time of imagine is not allowed (otherwise the handles HA would become invalid).

Examples:

1. `IMAGINE(DATA);`
   where DATA is a numeric or logical 3D-array starts IMAGINE with one panel displaying DATA. If DATA is a 4D-array, the data is split in N 3D-arrays, where N = size(DATA, 4) and displayed in a sufficient number of panels.
2. `IMAGINE(DATA, 'Titel');`
   Same functionality as above, however, supplying a title for the data which is displayed in the GUI. If DATA is a 4D-array, the titles are extended by a running index.
3. ```
   HA = IMAGINE(DATA);
   hold(HA(1), ‘on’);
   quiver(quiverdata_x, quiverdata_y, …);
   ```
   Overlays a quiver plot to the first axis in the IMAGINE UI.
   ```

Graphical Interface

Menubar

ctrl + o

Open File[1] Opens a standard file open dialog which can be used to open any file type that can be read using MATLAB's `imread` method. Images are sorted into series (which are displayed in different panels) according to image dimension.

Save Save the contents of the selected panels to image files (those supported by imwrite). Use the fieldnames %SeriesName% and %ImageNumber% in the filename which are automatically replaced with the corresponding variables to obtain different filenames for each image.

ctrl + i

Import Workspace Data[1] Allows importing variables from MATLAB's base workspace. Other workspaces (e.g. workspaces of functions in debug mode) are currently unsupported.

del

Delete Clears the content of the selected panel(s).

ctrl + x

Exchange Exchanges the contents of the two selected panels. Only available if exactly two panels are selected.

Grid Opens a pop-up window that allows choosing the amount of draw panels and the layout.

Colormap Chose between different predefined or custom colormaps. See below for further information about implementing your own colormaps.

Show/Hide Colorbar at the top of each panel. Use it to display the current dynamic range of each panel

Show/Hide Evalbar at the bottom of each panel. The evalbar can be used to show the results of evaluation functions (on the left) and shows the image value at the current cursor position for each panel.

ctrl + 0

Reset View Resets the zoom level to original size, windowing to full dynamic range and centers the images in their panels.

ctrl + l

Link Panels A toggle button that controls the UI's response to user input. When activated (default) all operations (e.g. scrolling, zooming, …) are applied to all visible panels. When inactive, actions are only applied to the selected panels. If no panel is selected, mouse actions are applied to the panel over which the mouse cursor is located.

---

[1] Note that adding data to the GUI doesn't change the panel layout, i.e. in order to see the newly added data, one has to increase the amount of panels or scroll through the data using the keyboard.

ctrl + p **Settings** Brings up a settings dialog that lets you select the background color and the evaluation functions.

Toolbar

The toolbar lets you select one of five different tools which are described in the following.

m **Pointer Tool** There are two types of operations: *clicking*- and *dragging* operations. *Click* the draw panels to **select/deselect** the corresponding panels or use in conjunction with the shift or control key to select multiple draw panels similar to selecting files in a file explorer.
Three types of *dragging* operations can be used: **Move** the images in their draw panels using the left mouse button, **zoom** images using the right mouse button and **window** (adjust brightness/contrast) the images using the middle mouse button.

r **Rotation Tool** Left-click and drag the mouse cursor in any direction. After a certain distance the image will **rotate**, with the initial point becoming the current slice. Comes in handy for 3D datasets. However, no interpolation will be performed. With this tool selected you can still perform the **zoom** and **window** function just like using the pointer tool. (Sorry for the icon, I didn't find anything more suitable in the icon set).

l **Line Profile Evaluation Tool** Use this tool to draw a **profile line** the current image. The line will automatically be drawn in all drawing panels. On releasing the mouse button, the underlying image data of all panels will be interpolated along the lines and sent to the line evaluation function together with the image captions. Interpolation is scaled such that the length of the data vector equals the length of the line in pixels. The line evaluation function can be selected using the *preferences* dialog. See the section *Writing Evaluation Functions* to find out how to implement evaluation functions for your needs.

o **ROI Evaluation Tool** Use this tool to create a **polygon ROI** in all panels. Left-clicking adds points to the polygon; right-clicking deletes the last polygon point. Close the polygon by either double-clicking or middle mouse button. Upon closing the polygon, the data will be sent to the ROI evaluation function which can be selected using the *preferences* dialog. See the section *Writing Evaluation Functions* to find out how to implement evaluation functions for your needs.

t **Tag Tool** Use this tool to **rename** panel data.

Keyboard Functions

Besides the menu shortcuts using the control key and the single letter shortcuts for the tool selection shown in the above tables there are some keyboard functions for scrolling images and series. Use the up/down arrows to navigate through the series and the left/right arrows to scroll through the image stacks. Those of you who are used to Siemens medical imaging consoles may find it convenient that the same functionality is found on the number block: Use the 4/5 keys to navigate through the series and the 1/2 keys to scroll through the image stacks.

Writing Evaluation Functions

IMQGINE currently supports two types to evaluation functions: Line evaluation and region of interest (ROI) evaluation functions. For convenience, these functions can be written in dedicated m-files and stored in the subdirectory “EvalFunctions”. The active evaluation function (i.e. the function that will be executed after having defined a line or ROI in the IMAGINE UI) can be set in the preferences dialog (ctrl + p). Both types have slightly different interfaces which are explained in the following.

Line Evaluation Functions

The IMAGINE UI calls a line evaluation function with the following syntax:

```
LINEEVALFCN(SData, sSelectionType hTexts);
```

The first argument `SData` is of type struct and has size Nx1, where N is the number of line profiles visible in IMAGINE. Each entry in `SData` contains two fields, `sName` containing a string with the caption of the corresponding series in IMAGINE and `dLineData` containing the corresponding line profile data. The data in `dLineData` is interpolated such that the distance between two entries is exactly on pixel in the original data, i.e. interpolation only is done if the profile lines are drawn olique.

The second argument `sSelectionType` is of type string and simply forwards the selection type of the IMAGINE figure which can be either ‘Normal’, ‘Extend’, ‘Alternate’ or ‘Open’. The meanings of these values depend on the OS and can be found in the MATLAB help page of the figure properties. This argument can be used to implement different evaluation function behavior depending on the keys pressed during the line drawing operation.

The third input is an array of handles (of uicontrol text components) of size Nx1. Use these handles to display evaluation information directly in the evalbar of the IMAGINE GUI.

See the supplied function *fEvalLineFWHM* for an example line profile evaluation function.

ROI Evaluation Functions

The IMAGINE UI calls a ROI evaluation function with the following syntax:

```
ROIEVALFCN(SData, hTexts);
```

The first argument `SData` is of type struct and has size Nx1, where N is the number of ROIs visible in IMAGINE. Each entry in `SData` contains three fields. `sName` contains a string with the caption of the corresponding series in IMAGINE. `dData` contains the intensities of all pixels within the ROI in a vector. The third field `lMask` is a binary representation of the the ROI and is the same for all entries in `SData`.

The second input is an array of handles (of uicontrol text components) of size Nx1. Use these handles to display evaluation information directly in the evalbar of the IMAGINE GUI.

See the supplied function *fEvalROIMean* for an example ROI evaluation function.

Implementing Custom Colormaps

To use your own customized colormap, simply create a function with the following interface:

```
Function dColormap = COLORMAPFCN(N);
```

This function should return a colormap array dColormap of size Nx3 with values in [0 1] and the column vectors representing red green and blue (just like the built-in colormap functions). Place this file in the sub-folder *colormaps* and and IMAGINE will automatically add it to the colormap options.

Contribute Yay!

Wanna make IMAGINE better? I see two ways of contributing to that. First, send your feedback/whishes/bug reports to christian.wuerslin@med.uni-tuebingen.de. Second, if you wrote an evaluation function that you think is awesome and you would like to share with the world, send it to the same address and I will incorporate it into the next release.

Revision History

V 1.0	02/21/13	Initial Version
V 1.1	02/25/13	- Auto-add eval function subdirectory to MATLAB path - Added tooltips
V 1.2	03/20/13	- Changed GUI architecture (now uses nested functions). - A lot of comments throughout the code. - Fixed a bug that occurred when deleting data. - Image value display automatically switches to exponential representation when showing values < 0.01. - Colors of the line/ROI evaluation functions now follow the MATLAB color order for better visual discrimination. - Rotation tool is now more robust. - Line profiles show captions. - New syntax can return the axes handles thus allowing the user to add plots to the axes. - Panel data can be exchanged.
V 1.3	04/03/13	- Added save function - Added App
V 1.4	04/24/13	- Added colormaps - Added colorbar option - Improved zooming operation - Added evalbar option