Readimx for Matlab: Overview

LAVISION

- reads all LaVision file formats: IMG, IMX, VEC, VC7
- gives access to all data in files
- provides data in an hierarchical structure
- avoids data conversion/copying

Readimx for Matlab: Download

- Our webpage: www.lavision.de (soon)
- Requirements: at least Matlab 2013a for 64bit platforms
  (Windows 7 / Mac OSX 10.8 / Linux)
- Windows 7: readimx4matlab_v2.0_2013a_w64.zip
- Mac OSX 10.8: readimx4matlab_v2.0_2013a_maci64.tar.bz2
- Linux: readimx4matlab_v2.0_2013a_glnx64.tar.bz2

Readimx: Installation (Windows 7)

- **Windows 7:**

  1. Unzip readimx4matlab_v2.0_2013a_w64.zip in folder of your choice
     (e.g. C:\Users\MY_ACCOUNT\Documents\MATLAB\readimx)

  2. Open Matlab and add the path to the search path
     (e.g***. >>addpath <MY_READIMX_PATH>*** or use ***>>pathtool*** command)

  3. Run demo (***>>readimxdemo***)

Readimx: Installation (Mac OS X ≥10.8)

- **Mac OS X ≥10.8:**

1. Unpack readimx4matlab_v2.0_2013a_maci64.tar.gz in an installation folder of your choice (e.g. /Users/MY_ACCOUNT/Documents/MATLAB/readimx)

2. The readimx package needs an extension of the environment variable DYLD_LIBRARY_PATH. Therefore, we provide an *AppleScript* file and a *bash* file to start Matlab in this specific manner. The files can be found in the sub-folder *Tools* of the package. They need to be adopted to your specific settings with an simple text editor.

3. Open Matlab with the script and add the installation folder to Matlab's search path (e.g. ***>>addpath <MY_READIMX_PATH>*** or use ***>>pathtool*** command)

4. Run demo (***>>readimxdemo***)

Readimx: Installation (Linux)

- **Linux:**

1. Unpack readimx4matlab_v2.0_2013a_glnxa64.tar.gz in an installation folder of your choice (e.g. /Users/MY_ACCOUNT/Documents/MATLAB/readimx)

2. The readimx package needs an extension of the environment variable LD_LIBRARY_PATH. Therefore, we provide a *bash* file
to start Matlab in this specific manner. The file can be found in the sub-folder *Tools*
of the package. They need to be adopted to your specific settings with an simple text editor.

3. Due to problems *to libstdc*++ problems, you have to change your Matlab installation (please ask your administrator for help):

   1. open a terminal
   2. execute “cd <MATLAB_BIN>/sys/os/glnxa64”, (usually <MATLAB_BIN> is /usr/local/MATLAB/R20XX/)
   3. execute ‘”sudo mkdir LV_Backup”
   4. execute “sudo mv libstdc++* LV_Backup/”

4. Open Matlab with the script and add the installation folder to Matlab’s search path (e.g. ***>>addpath <MY_READIMX_PATH>*** or use ***>>pathtool*** command)

5. Run demo (***>>readimxdemo***)

Readimx: Buffer Data Model

Buffer-Model

The Matlab command

>> B=readimx('C:/temp/B00001.im7') % Read buffer from a file

loads an image file and creates a **Buffer** structure with two fields:

```
B =
    Frames: {[1x1 struct]}
    Attributes: {19x1 cell}
```

The field *Frames* is a cell array of **Frame** structures and holds all frame data of the file.

The field *Attributes* is a cell array of **Attribute** structure and holds all buffer attributes from the file.

Readimx: Frame Data Model

Frame-Model

The Matlab command

```
>>F=B.Frames{1}    % Access 1. frame
```

returns a **Frame** structure with the following fields:

```
F=
   Components: {[1x1 struct]}
   Attributes: {13x1 cell}
   Scales: [1x1 struct]
   ComponentNames: {'PIXEL'}
   IsVector: 0
   Grids: [1x1 struct]
```

The field *Components is a cell array of* **Component** structures and holds the image or vector data.
The field *Attributes* is a cell array of **Attribute** structures an hold the frame attributes.
The field *Scales* is a structure holding the X, Y, Z ,I scale information of the image or vector data.
The field *ComponentNames* is a cell array of names, giving the meanings of the components.
The field *IsVector* indicates weather the frame contains vector components or image planes.
The field *Grids* is a structure for grid spaces in X, Y, Z direction.

Readimx: Component Data Model

Component-Model

The Matlab command

>>C=B.Frames{1}.Components{1} }   % Access 1. component of 1. frame

returns a **Component** structure with the following fields:

```
C=
  Scale: [1x1 struct]
  Planes: {[1152x896 uint16]}
  Name: 'PIXEL‘
```

The field *Scale is a* **Scale** structure for intensity scaling of image or component data.

The field *Planes* is a cell array of **Plane** structures and holds the components data.

The field *Name* is the component name of this component.

Readimx: Other Data Models

A **Plane** is a 2-dimensional data array containing data for image (planes) for vector (plane) components.

The **Scale** structure provides data for linear mapping plane data to physical quantities. It has the following fields: *Slope, Offset, Unit, Description.*
The mapping is done by $f(I) = A*I + B$, with slope *A* and offset *B*. *Unit* and *Description* are simple string.

The **Attribute** structure has two fields: *Name, Value*. The *Name* field is always a string and gives the attribute an identifier. The *Value* field hold the attribute data and has different type: *double, string, array*.

Readimx: Scripts

The delivered script files gives an example and a template for the evaluation of readimx results (buffer).

File	Description
showimx.m	get a frame, display the data, return compiled data
show2DVec.m	get a frame, display the vectors, return compiled 2D vector data
show3DVec.m	get a frame, display the vectors, return compiled 3D vector data
showPlane.m	get plane and scales, display the image, return compiled 2D image data
showVolume.m	get plane and scales, display slices, return compiled 3D image data
create2DVec.m	get a frame, return compiled 2D vector data
create3DVec.m	get a frame, return compiled 3D vector data
createPlane.m	get plane and scales, return compiled 2D image data
createVolume.m	get plane and scales, return compiled 3D image data
makeFrameInfo.m	get frame, return frame information
readimxdemo.m	read demo files, display results