kermit

Generated by Doxygen 1.8.11

Contents

Chapter 1

Namespace Index

1.1 Namespace List

Here is a list of all namespaces with brief descriptions:

Chapter 2

Hierarchical Index

2.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

Chapter 3

Class Index

3.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

Chapter 4

File Index

4.1 File List

Here is a list of all files with brief descriptions:

Chapter 5

Namespace Documentation

5.1 kermit Namespace Reference

Classes

- class KerrArray
- class KerrGUI

Variables

- tuple GRAY_RANGE = (0,65535)
- tuple IM_SIZE = (512,672)
- tuple AN_IM_SIZE = (554,672)
- tuple StringTypes = (str,unicode)
- string ex_data1 = 'ExampleData1'
- string ex_data2 = 'ExampleData2'
- string tmp_dir = 'tmp'
- string example_im_fol = r'C:\Users\phyrct\Dropbox\Me\Coding\kermit'
- bkim = io.imread('bknd.png')
- unpim = io.imread('unpro.png')
- im = io.imread('sub.png')
- list proc_list = [im]
- v1 = CollectionViewer(proc_list)

5.1.1 Variable Documentation

5.1.1.1 tuple kermit.AN_IM_SIZE = (554,672)

5.1.1.2 kermit.bkim = io.imread('bknd.png')

5.1.1.3 string kermit.ex_data1 = 'ExampleData1'

5.1.1.4 string kermit.ex_data2 = 'ExampleData2'

5.1.1.5 string kermit.example_im_fol = r'C:\Users\phyrct\Dropbox\Me\Coding\kermit'

5.1.1.6 tuple kermit.GRAY_RANGE = (0,65535)

5.1.1.7 kermit.im = io.imread('sub.png')

5.1.1.8 tuple kermit.IM_SIZE = (512,672)

5.1.1.9 list kermit.proc_list = [im]

5.1.1.10 tuple kermit.StringTypes = (str,unicode)

5.1.1.11 string kermit.tmp_dir = 'tmp'

5.1.1.12 kermit.unpim = io.imread('unpro.png')

5.1.1.13 kermit.v1 = CollectionViewer(proc_list)

5.2 kerrlist Namespace Reference

Classes

- class KerrList

Variables

- tuple GRAY_RANGE = (0,65535)
- tuple IM_SIZE = (512,672)
- tuple AN_IM_SIZE = (554,672)
- tuple StringTypes = (str,unicode)

5.2.1 Variable Documentation

5.2.1.1 tuple kerrlist.AN_IM_SIZE = (554,672)

5.2.1.2 tuple kerrlist.GRAY_RANGE = (0,65535)

5.2.1.3 tuple kerrlist.IM_SIZE = (512,672)

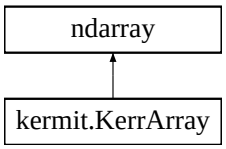
5.2.1.4 tuple kerrlist.StringTypes = (str,unicode)

Chapter 6

Class Documentation

6.1 kermit.KerrArray Class Reference

Inheritance diagram for kermit.KerrArray:



Public Member Functions

- def __new__ (cls, image, metadata={})
- def __init__ (self)
- def __array_finalize__ (self, obj)
- def __array_wrap__ (self, out_arr, context=None)
- def crop_text (self, copy=False)
- def crop_image (self, coord=None, copy=True)
- def level_image (self, poly_vert=1, poly_horiz=1, box=None, poly=None)
- def get_metadata (self, field_only=False)
- def trace (self, start, end, width=1, order=1)
- def filter_image (self, sigma=2, box=None)
- def translate (self, translation)
- def rotate (self, rotation)
- def translate_limits (self, translation)
- def split_image (self)
- def edge_det (filename, threshold1, threshold2)
- def NPPixel_BW (np_image, thresh1, thresh2)

Public Attributes

- metadata
- shape

6.1.1 Detailed Description

```
Class for manipulating Kerr images from Evico software.
It is built to be almost identical to a numpy array except for one extra
parameter which is the metadata. This stores information about the image
in a dictionary object for later retrieval.
All standard numpy functions should work as normal and casting two types
together should yield a KerrArray type (ie. KerrArray+np.ndarray=KerrArray)

A note on coordinate systems:
For arrays the indexing is (row, column). However the normal way to index
an image would be to do (horizontal, vert), which is the opposite.
In KerrArray the coordinate system is chosen similar to skimage. y points
down x points right and the origin is in the top left corner of the image.
When indexing the array therefore you need to give it (y,x) coordinates
for (row, column).

 ----> x (column)
|
|
v
y (row)

eg I want the 4th pixel in the horizontal direction and the 10th pixel down
from the top I would ask for KerrArray[10,4]
```

6.1.2 Constructor & Destructor Documentation

6.1.2.1 def kermit.KerrArray.__init__ (*self*)

```
called by __new__ when it has finished
```

6.1.3 Member Function Documentation

6.1.3.1 def kermit.KerrArray.__array_finalize__ (*self, obj*)

```
__array_finalize__ and __array_wrap__ are necessary functions when
subclassing numpy.ndarray to fix some behaviours. See
http://docs.scipy.org/doc/numpy-1.10.1/user/basics.subclassing.html for
more info and examples
```

6.1.3.2 def kermit.KerrArray.__array_wrap__ (*self, out_arr, context* = None)

```
see __array_finalize__ for info
```

6.1.3.3 def kermit.KerrArray.__new__ (*cls, image, metadata* = {})

```
Construct a Kermit object. We're using __new__ rather than __init__
to imitate a numpy array as close as possible.

Parameters
----------
image: string or numpy array initiator
    If a filename is given it will try to load the image from memory
    Otherwise it will call np.array(image) on the object so an array or
    list is suitable
metadata: dict
    dictionary of metadata items you would like adding to your array
Returns
-------
ka: KerrArray
    A KerrArray object with metadata attached
```

6.1.3.4 def kermit.KerrArray.crop_image (*self, coord =* None, *copy =* True)

```
Crop the image.
Crops to the coord given or defaults to allowing the user
to draw a rectangle. Returns the cropped image.

Parameters
----------
coord: array or list of type int:
    [xmin,xmax,ymin,ymax]
copy: bool
    whether to return a copy of the array or a view of the original object

Returns
-------
im: KerrArray
    cropped image
```

6.1.3.5 def kermit.KerrArray.crop_text (*self, copy =* False)

```
Crop the bottom text area from a standard Kermit image

Parameters
----------
copy: bool
    Whether to return a copy of the data or the original data

Returns
-------
im: KerrArray
    cropped image
```

6.1.3.6 def kermit.KerrArray.edge_det (*filename, threshold1, threshold2*)

```
Detects an edges in an image according to the thresholds 1 and 2.
Below threshold 1, a pixel is disregarded from the edge
Above threshold 2, pixels contribute to the edge
Inbetween 1&2, if the pixel is connected to similar pixels then the pixel conributes to the edge
```

6.1.3.7 def kermit.KerrArray.filter_image (*self, sigma =* 2, *box =* None)

```
Apply a filter to an area of the image defined by box
call through to skimage.filters.gaussian

Parameters
----------
sigma: float
    standard deviation for gaussian blur
box: 4-tuple
    area to apply blur to (xmin,xmax,ymin,ymax)

Returns
-------
image
    filtered image
```

6.1.3.8 def kermit.KerrArray.get_metadata (*self, field_only =* `False`)

```
Use image recognition to try to pull the metadata numbers off the image

Requirements: This function uses tesseract to recognise the image, therefore
tesseract file1 file2 must be valid on your command line.
Install tesseract from
https://sourceforge.net/projects/tesseract-ocr-alt/files/?source=navbar

Parameters
----------
field_only: bool
    only try to return a field value

Returns
-------
metadata: dict
    updated metadata dictionary
```

6.1.3.9 def kermit.KerrArray.level_image (*self, poly_vert =* `1`, *poly_horiz =* `1`, *box =* `None`, *poly =* `None`)

```
Subtract a polynomial background from image

Fit and subtract a background to the image. Fits a polynomial of order
given in the horizontal and vertical directions and subtracts. If box
is defined then level the *entire* image according to the
gradient within the box.

Parameters
----------
poly_vert: int
    fit a polynomial in the vertical direction for the image of order
    given. If 0 do not fit or subtract in the vertical direction
poly_horiz: int
    fit a polynomial of order poly_horiz to the image. If 0 given
    do not subtract
box: array, list or tuple of int
    [xmin,xmax,ymin,ymax] define region for fitting. IF None use entire
    image
poly: list or None
    [pvert, phoriz] pvert and phoriz are arrays of polynomial coefficients
    (highest power first) to subtract in the horizontal and vertical
    directions. If None function defaults to fitting its own polynomial.

Returns
-------
im: KerrArray
    the levelled image
```

6.1.3.10 def kermit.KerrArray.NPPixel_BW (*np_image, thresh1, thresh2*)

```
Changes the colour if pixels in a np array according to an inputted threshold
```

6.1.3.11 def kermit.KerrArray.rotate (*self, rotation*)

```
Rotates the image.
Areas lost by move are cropped, and areas gained are made black (0)

Parameters
----------
rotation: float
    clockwise rotation angle in radians (rotated about top right corner)

Returns
-------
im: KerrArray
    rotated image
```

6.1.3.12 def kermit.KerrArray.split_image (*self*)

```
split image into different domains, maybe by peak fitting the histogram?
```

6.1.3.13 def kermit.KerrArray.trace (*self, start, end, width* = 1, *order* = 1)

```
Return a line trace of intensity averaging over width.
Call through to skimage.measure.profile_line

Parameters
----------
start: 2-tuple
    coords at start of line (numpy coordinates not x,y)
end: 2-tuple
    coords at end of line (last pixel is included in result)
width: int
    width of line to average over
order: int
    order of the spline interpolation

Returns
-------
profile: array
    intensity profile
```

6.1.3.14 def kermit.KerrArray.translate (*self, translation*)

```
Translates the image.
Areas lost by move are cropped, and areas gained are made black (0)

Parameters
----------
translate: 2-tuple
    translation (x,y)

Returns
-------
im: KerrArray
    translated image
```

6.1.3.15 def kermit.KerrArray.translate_limits (*self, translation*)

```
Find the limits of an image after a translation
After using KerrArray.translate some areas will be black,
this finds the area that still has original pixels in

Parameters
----------
translation: 2 tuple
    the (x,y) translation applied to the image

Returns
-------
limits: 4-tuple
    (xmin,xmax,ymin,ymax
```

6.1.4 Member Data Documentation

6.1.4.1 kermit.KerrArray.metadata

6.1.4.2 kermit.KerrArray.shape

The documentation for this class was generated from the following file:

- C:/Users/phyrct/Dropbox/Me/Coding/kermit/kermit/kermit.py

6.2 kermit.KerrGUI Class Reference

Public Member Functions

- def draw_rectangle (self)
- def draw_trace (vert_coord, width=1)
- def plt_histogram (self, kwarg)

6.2.1 Member Function Documentation

6.2.1.1 def kermit.KerrGUI.draw_rectangle (*self*)

```
Draw a rectangle on the image and return the coordinates

Returns
-------
coord: ndarray
    [xmin,xmax,ymin,ymax]
```

6.2.1.2 def kermit.KerrGUI.draw_trace (*vert_coord, width* = 1)

```
Line trace horizontal at vertical coord averaging over width
```

6.2.1.3 def kermit.KerrGUI.plt_histogram (*self, kwarg*)

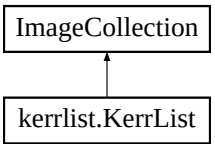
```
plot histogram of image intensities, pass through kwarg to matplotlib.pyplot.hist
```

The documentation for this class was generated from the following file:

- C:/Users/phyrct/Dropbox/Me/Coding/kermit/kermit/kermit.py

6.3 kerrlist.KerrList Class Reference

Inheritance diagram for kerrlist.KerrList:



Public Member Functions

- def __init__ (self, load_pattern, conserve_memory=True, load_func=None, load_func_kwargs)
- def hysteresis_loop (self, fieldlist=None, box=None)
- def drift_loop_correct (hysloop, manual=False)
- def faraday_correct (hysloop, manual=False)
- def correct_image_drift (self, ref, imlist, threshold=0.005)
- def transform_images (imlist, translation=None, rotation=None)

6.3.1 Detailed Description

```
KerrList groups functions that can be applied to a group of KerrImages.
In general it is designed to behave pretty much like a normal python list.
```

6.3.2 Constructor & Destructor Documentation

6.3.2.1 def kerrlist.KerrList.__init__ (*self, load_pattern, conserve_memory* = `True`, *load_func* = `None`, *load_func_kwargs*)

```
Initialise a KerrList. A list of images to manipulate. Mostly a pass
through to the skimage.io.ImageCollection class

Parameters
----------
load_pattern: str or list
    pattern of filenames to load. Uses standard glob nomenclature. Seperate
    different requests with a : eg 'myimages/*.png:myimages/*.jpg'. If
    a list is given it treats it as a list of image arrays.
pattern: str or list
    loading pattern with standard glob nomenclature (* wildcards,
    [0-9] character in this range etc.)
conserve_memory: bool
    parameter passed onto skimage.io.ImageCollection. Option for loading
    all the files into memory initially or later

Other parameters
----------------
load_func: callable or None
    see skimage.io.ImageCollection for notes.

Attributes
----------
files: list or str
    list of loaded file names. Or equal to load_pattern if a list was
    given.
```

6.3.3 Member Function Documentation

6.3.3.1 def kerrlist.KerrList.correct_image_drift (*self, ref, imlist, threshold* = 0.005)

```
Align images to correct for image drift.
Detects common features on the images and tracks them moving.

Parameters
----------
ref: np.ndarry
    reference image with zero drift
imlist: list or tuple of images
    images to find drift
threshold: float
    threshold for detecting imperfections in images

Returns
-------
shifts: array
    shift vector for each image in imlist relative to ref (x drift, y drift)
transim: list
    list of images with correct shifts applied
```

6.3.3.2 def kerrlist.KerrList.drift_loop_correct (*hysloop, manual* = False)

```
correct a linear drift in time on a hysteresis loop
```

6.3.3.3 def kerrlist.KerrList.faraday_correct (*hysloop, manual* = False)

```
correct for the faraday effect
```

6.3.3.4 def kerrlist.KerrList.hysteresis_loop (*self, fieldlist* = None, *box* = None)

```
Make a hysteresis loop of the average intensity in the given images

Parameters
----------
fieldlist: list or tuple
    list of fields used, if None it will try to get field from imgae metadata
box: list
    [xmin,xmax,ymin,ymax] region of interest for hysteresis
```

6.3.3.5 def kerrlist.KerrList.transform_images (*imlist, translation* = None, *rotation* = None)

```
Translate or rotate image or images.
Translates or rotates the images in the x-y plane. Areas lost by move are cropped, and
areas gained are made black.

Parameters
----------
im: array or list
    image or list of images to be translated
tranlations: tuple or list
    array of relative distances for translation [horizontal, vert]
    eg. [[1,3],[-5,4],[9,-8]]. Defaults to no translation
```

```
rotations: float or list
    list of rotation angles in radians to apply to the images (rotates about top left
    corner). Defaults to 0.

Returns
newims: list
    The transformed images (all of the same shape as the originals)
lims: array
    The limits of the image that have not been destroyed by translations
    [xmin,xmax,ymin,ymax] (doesn't account for rotation yet!)
```

The documentation for this class was generated from the following file:

- C:/Users/phyrct/Dropbox/Me/Coding/kermit/kermit/kerrlist.py

Chapter 7

File Documentation

7.1 C:/Users/phyrct/Dropbox/Me/Coding/kermit/kermit/kermit.py File Reference

Classes

- class kermit.KerrArray
- class kermit.KerrGUI

Namespaces

- kermit

Variables

- tuple kermit.GRAY_RANGE = (0,65535)
- tuple kermit.IM_SIZE = (512,672)
- tuple kermit.AN_IM_SIZE = (554,672)
- tuple kermit.StringTypes = (str,unicode)
- string kermit.ex_data1 = 'ExampleData1'
- string kermit.ex_data2 = 'ExampleData2'
- string kermit.tmp_dir = 'tmp'
- string kermit.example_im_fol = r'C:\Users\phyrct\Dropbox\Me\Coding\kermit'
- kermit.bkim = io.imread('bknd.png')
- kermit.unpim = io.imread('unpro.png')
- kermit.im = io.imread('sub.png')
- list kermit.proc_list = [im]
- kermit.v1 = CollectionViewer(proc_list)

7.2 C:/Users/phyrct/Dropbox/Me/Coding/kermit/kermit/kerrlist.py File Reference

Classes

- class kerrlist.KerrList

Namespaces

- kerrlist

Variables

- tuple kerrlist.GRAY_RANGE = (0,65535)
- tuple kerrlist.IM_SIZE = (512,672)
- tuple kerrlist.AN_IM_SIZE = (554,672)
- tuple kerrlist.StringTypes = (str,unicode)