xripl Documentation

Release 0.2.0

Pawel Marek Kozlowski

Jun 01, 2022

FIRST STEPS

1 Installing XRIPL 3

2 Examples 5

3 How to Contribute 7

4 Acknowledging and Citing 9

5 XRIPL License (BSD 3-clause) 11

6 Clean (*xripl.clean*) 13

7 Contour (*xripl.contour*) 15

8 Contrast (*xripl.contrast*) 17

9 Films (*xripl.films*) 19

10 Filters (*xripl.filters*) 21

11 Instrument (*xripl.instrument*) 23

12 Magnification (*xripl.magnification*) 25

13 Plot Defaults (*xripl.pltDefaults*) 29

14 Reader (*xripl.reader*) 31

15 Segmentation (*xripl.segmentation*) 33

16 Tube Transmission (*xripl.tubeTransmission*) 37

17 Visualizations (*xripl.visualizations*) 41

18 Indices and tables 45

Python Module Index 47

Index 49

XRIPL (read as "zripple") is a library of tools for processing x-ray radiographs and extracting contours of interest within the image using computer vision techniques. Features include spatial calibration, denoising, background and attenuation correction through pseudo-flatfielding, watershed segmentation, contour processing, and visualization. The details of the XRIPL analysis pipeline are published in the Proceedings of the 23rd Topical Conference on High-Temperature Plasma Diagnostics Proceedings [1].

[1] P. M. Kozlowski, Y. Kim, B. M. Haines, H. F. Robey, T. J. Murphy, H. M. Johns, and T. S. Perry. Use of Computer Vision for analysis of image data sets from high temperature plasma experiments. Review of Scientific Instruments 92, 033532 (2021) https://doi.org/10.1063/5.0040285

CHAPTER

ONE

INSTALLING XRIPL

1.1 Requirements

XRIPL requires Python version 3.7 or newer. XRIPL also require the following openly available packages for installation:

- NumPy — 1.15.0 or newer
- matplotlib — 2.2.2 or newer
- scipy — 1.1.0 or newer
- h5py — 2.8.0 or newer
- scikit-image — 0.14.0 or newer

1.2 Installation with pip

Official releases of XRIPL are published to pypi.org and can simply be pip installed like so:

```
pip install xripl
```

1.3 Building and installing from source (for contributors)

1.3.1 Make sure you have python installed, preferably via Anaconda

Here is where you get Anaconda, and make sure to get the Python 3 version. https://www.anaconda.com/distribution/

1.3.2 Setup installation directory

Make a directory called "xripl" in a sensible place on your system. Preferably in a directory where none of the higher level directory names have spaces in them.

1.3.3 Setup a virtual environment

If you have python installed via Anaconda, then create your virtual environment like this

```
conda create --name xripl
```

1.3.4 Clone the repository using git

In the xripl directory you created, run the following on the command line

```
git clone https://github.com/lanl/xripl.git
```

1.3.5 Activate your virtual environment

Still on the command line, run

```
source activate xripl
```

1.3.6 Install requirements

```
pip install -r requirements.txt
```

1.3.7 Install xripl

If you are a user then do

```
pip install .
```

If you wish to help in developing xripl, then do

```
pip install -e .
```

1.3.8 Test if install was successful

Open a python and try doing `import xripl`. If all went well then you shouldn't get any error messages.

CHAPTER

TWO

EXAMPLES

2.1 General examples

General-purpose and introductory examples from XRIPL

2.1.1 Radiograph segmentation tutorial

Importing modules

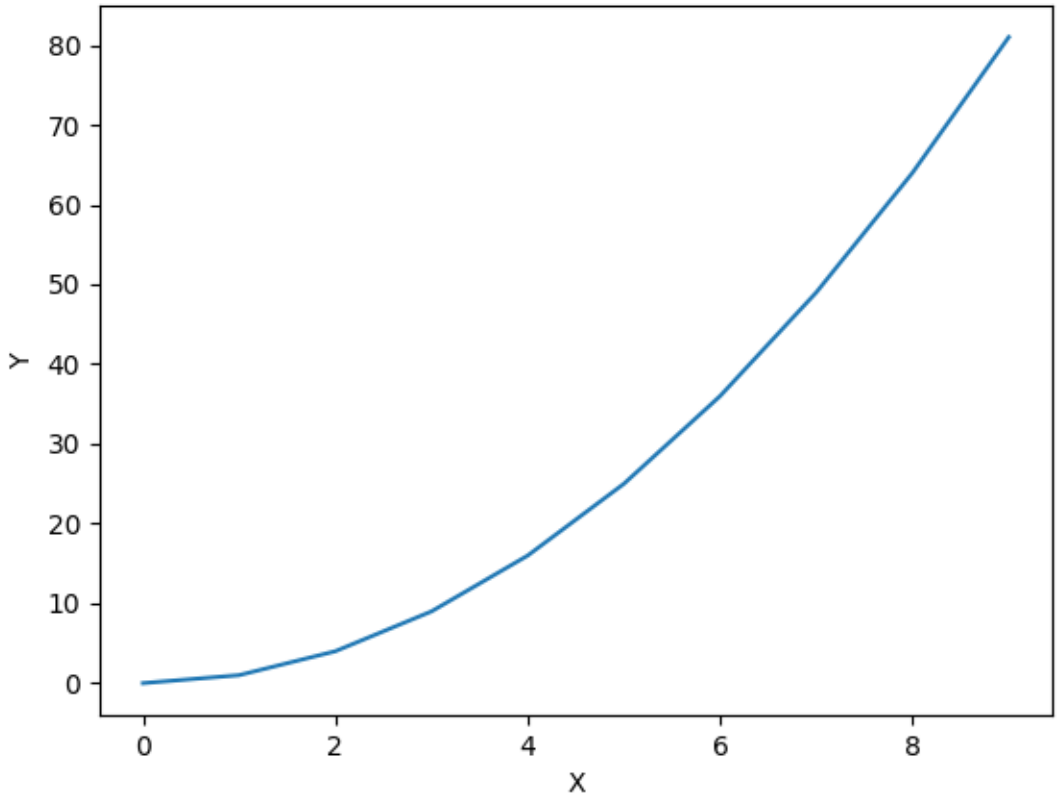
```
import numpy as np
import matplotlib.pyplot as plt
from skimage.filters import median, rank

#from xripl.data import shot
from xripl.reader import openRadiograph
from xripl.contrast import equalize
from xripl.clean import cleanArtifacts, flatten
from xripl.segmentation import detectShock
# import xripl.pltDefaults
```

Plot

```
xData = np.arange(10)
yData = xData ** 2

plt.plot(xData, yData)
plt.xlabel('X')
plt.ylabel('Y')
plt.show()
```



Total running time of the script: (0 minutes 0.095 seconds)

CHAPTER

THREE

HOW TO CONTRIBUTE

Visit our GitHub repository and look through the list of open issues to see how you can contribute. If you find a bug, or would like to see a feature enhancement, then open up an issue and describe it detail.

CHAPTER

FOUR

ACKNOWLEDGING AND CITING

If you use XRIPL for work/research presented in a publication (whether directly, or as a dependency to another package), we encourage the following acknowledgement:

> This research made use of XRIPL, a community-developed Python package for analysis of filtered diode array signals.

and that you cite the following paper(s):

```
@article{kozlowski2021use,
  title={Use of computer vision for analysis of image datasets from high temperature␣
↪plasma experiments},
  author={Kozlowski, Pawel Marek and Kim, Y and Haines, Brian Michael and Robey, HF and␣
↪Murphy, Thomas Joseph and Johns, Heather Marie and Perry, Theodore Sonne},
  journal={Review of Scientific Instruments},
  volume={92},
  number={3},
  pages={033532},
  year={2021},
  publisher={AIP Publishing LLC}
}
```

CHAPTER

FIVE

XRIPL LICENSE (BSD 3-CLAUSE)

CHAPTER

SIX

CLEAN (*XRIPL.CLEAN*)

Created on Thu Aug 1 13:13:52 2019

Functions for cleaning and regularizing images.

@author: Pawel M. Kozlowski

6.1 Functions

cleanArtifacts(image[, diskSize, plots, ...])	Applies morphological opening to clean bright streaks and spots from radiographs, and then applies a morphological closing to remove dark artifacts.
flatten(image[, medianDisk, gaussSize, ...])	Produces an approximate image of the background light intensity variations in the image and divides the image by this background to effectively flatten the contrast.
enhanceRadiograph(img[, medianDisk, ...])	Median filter, morphological filter, and pseudo-flatfield the radiographs to prepare it for feature identification.

6.1.1 cleanArtifacts

xripl.clean.**cleanArtifacts**(*image*, *diskSize=5*, *plots=False*, *vmin=0*, *vmax=1*, *flip=False*)

Applies morphological opening to clean bright streaks and spots from radiographs, and then applies a morphological closing to remove dark artifacts.

Parameters

- **image** (*numpy.ndarray*) – Image containing artifacts to be cleaned.
- **diskSize** (*int*) – Size of disk to be used as convolving element for removing artifcats.
- **plots** (*bool*) – Flag for displaying plots of processed images.
- **vmin** (*float*) – Value between 0 and 1 for setting minimum threshold in plotting image. See matplotlib.pyplot.imshow().
- **vmax** (*float*) – Value between 0 and 1 for setting maximum threshold in plotting image. See matplotlib.pyplot.imshow().
- **flip** (*bool*) – Flips order of opening and closing operations. When False, opening is applied first, then closing. When True, closing is applied first, then opening. False is used by default for radiographs, and True is used for inverted radiographs

6.1.2 flatten

xripl.clean.**flatten**(*image*, *medianDisk=100*, *gaussSize=100*, *plots=False*, *vmin=0*, *vmax=1*)

Produces an approximate image of the background light intensity variations in the image and divides the image by this background to effectively flatten the contrast.

Parameters

- **image** (*numpy.ndarray*) – Image whose background brightness variation is to be removed via flattening.
- **medianDisk** (*int*) – Size of disk used in median filtering to smooth out all structures in image other than background light variations.
- **gaussSize** (*int*) – Size of Gaussian smoothing used on background image produced by median filtering. This smooths out ridges that appear in the median filtered image.
- **plots** (*bool*) – Flag for displaying plots of processed images.

6.1.3 enhanceRadiograph

xripl.clean.**enhanceRadiograph**(*img*, *medianDisk=5*, *morphDisk=5*, *flattenMedian=100*, *flattenGauss=100*, *plots=False*)

Median filter, morphological filter, and pseudo-flatfield the radiographs to prepare it for feature identification.

Parameters

- **img** (*numpy.ndarray*) – Foreground radiographic image as a 2D numpy array, which is to be cleaned/enhanced.
- **medianDisk** (*int*) – Radial size of disk in pixels to be used for median filtering the image. Default is radius of 5 pixels.
- **morphDisk** (*int*) – Radial size of disk in pixels to be used for morphologically filtering the image to clean up artifacts.
- **flattenMedian** (*int*) – Radial size of disk in pixels to be used for median filtering image as part of over-smoothing process to obtain the pseudo-flatfield. Default is radius of 100 pixels.
- **flattenGauss** (*int*) – Standard deviation in pixels to be used for Gaussian blurring the image as part of over-smoothing process to obtain the pseudo-flatfield. Default is sigma of 100 pixels.
- **plots** (*bool*) – Flag for plotting cleaned and flattened images. Default is False.

CHAPTER

SEVEN

CONTOUR (*XRIPL.CONTOUR*)

Created on Mon Oct 22 16:47:46 2018

Tools for identifying contours of equal intensity/contrast.

@author: Pawel M. Kozlowski

7.1 Functions

nContours(contours, n)	Get the n longest contours from a list of contours.

7.1.1 nContours

xripl.contour.**nContours**(*contours*, *n*)

Get the n longest contours from a list of contours.

contours: list
A list of contours from skimage.measure.find_contours().

n: int
Number of contours to select.

CHAPTER

EIGHT

CONTRAST (*XRIPL.CONTRAST*)

Created on Mon Oct 22 16:46:08 2018

Functions for adjusting plotting and adjusting the contrast of radiographic images.

@author: Pawel M. Kozlowski

8.1 Functions

equalize(image[, plot, savePlot, fileName, ...])	Convenience function for applying contrast equalization to radiographic data.

8.1.1 equalize

xripl.contrast.**equalize**(*image*, *plot=False*, *savePlot=False*, *fileName=''*, *saveDir=''*, *cmap='viridis'*, *kernel_size=(256, 256)*, *clip_limit=0.03*, *nbins=256*)

Convenience function for applying contrast equalization to radiographic data. Applies both global contrast equalization and CLAHE equalization.

CHAPTER

NINE

FILMS (*XRIPL.FILMS*)

Created on Wed Feb 27 15:32:34 2019

Opening and processing microD scanned films.

@author: Pawel M. Kozlowski

9.1 Functions

openFilm(fileName)	Opens HDF5 file film scan from microD.

9.1.1 openFilm

xripl.films.**openFilm**(*fileName*)

Opens HDF5 file film scan from microD.

CHAPTER

TEN

FILTERS (*XRIPL.FILTERS*)

Created on Tue Mar 31 09:50:47 2020

Custom filters for cleaning/blurring images.

@author: Pawel M. Kozlowski

10.1 Functions

boxFilter(img[, boxSize])	Applies a symmetric box filter to blur the image.

10.1.1 boxFilter

xripl.filters.**boxFilter**(*img*, *boxSize=2*)

Applies a symmetric box filter to blur the image. A box filter approaches a Gaussian filter for large number of filterings, but at smaller kernel sizes it causes less blur. Convolution is handled using scipy.ndimage.convolve, and edges condition is 'reflect'.

img
: [numpy.ndarray] 2D numpy array of the image to be filtered.

boxSize
: [int] Length in pixels of the side of the box. The box is symmetric. Default size 2 pixels by 2 pixels.

filteredImg
: [numpy.ndarray] Box filtered image returned as a 2D numpy array.

CHAPTER

ELEVEN

INSTRUMENT (*XRIPL.INSTRUMENT*)

Created on Tue Feb 26 12:58:48 2019

Utilities for convolving 2D radiograph instrument function with synthetically produced radiographs.

@author: Pawel M. Kozlowski

11.1 Functions

openSyntheticRadiograph(directory, fileName)	Given the full filename (with path) to an hdf5 file containing synthetic radiograph data, open the file and return the image data as a numpy array.
saveHdf(data, directory, fileName)	Saves radiograph into HDF5 file.

11.1.1 openSyntheticRadiograph

xripl.instrument.**openSyntheticRadiograph**(*directory*, *fileName*, *plot=False*)

Given the full filename (with path) to an hdf5 file containing synthetic radiograph data, open the file and return the image data as a numpy array.

fileName is a str

11.1.2 saveHdf

xripl.instrument.**saveHdf**(*data*, *directory*, *fileName*)

Saves radiograph into HDF5 file.

CHAPTER

TWELVE

MAGNIFICATION (*XRIPL.MAGNIFICATION*)

Created on Wed Jun 3 14:51:45 2020

@author: Pawel M. Kozlowski

12.1 Functions

lineoutMinima(lineoutX, lineoutY[, ...])	Given lineout should be lineout from flattened radiograph image.
tubeCenterPx(minimaIdxs[, lineoutY, plotsFlag])	Given the pixel positions of the tube edges, obtains the tube center in pixels for a tube target.
diameterPx(minimaIdxs)	Obtains the diameter of the tube in pixels when given the tube edges.
magnificationUmPx(innerDiameterPx[, ...])	Get magnification with propagated uncertainty in um/px when given target tube diameter in um, in px, and standard deviation on the diameter in px.
magnificationAnalysis(shot, camera, dataDir, ...)	Get magnification from target radiograph by measuring inner tube wall of known diameter.

12.1.1 lineoutMinima

xripl.magnification.**lineoutMinima**(*lineoutX*, *lineoutY*, *window_length=51*, *polyOrder=3*, *peakWidth=50*, *plotsFlag=False*)

Given lineout should be lineout from flattened radiograph image. This function smoothes the lineout using a Savitzky-Golay filter, and finds the 2 deepest minima in the lineout. These minima should correspond to the inner diameter edge of the target tube.

Parameters

- **lineoutX** (*numpy.ndarray*) – 1D array of x-axis points of the lineout from which we want to get minima.
- **lineoutY** (*numpy.ndarray*) – 1D array of y-axis points of the lineout from which we want to get minima. This should be a radial lineout through the tube target. Finds minima in intensity, which should correpsond to the inner diameter edge of the tube.
- **window_length** (*int*) – Window length for Savitzky-Golay filter. Must be odd numbered. Default is window that is 51 points long.
- **polyOrder** (*int*) – Polynomial order for Savitzky-Golay filter. Default is 3 (cubic).

- **peakWidth** (*int*) – Minimum number of points required in peak width to select the peak as a minima. Default is 50 pts wide.
- **plotsFlag** (*bool*) – Flag for plotting smoothed lineout over input lineout. Also plots minima points over smoothed lineout. Default is False.

12.1.2 tubeCenterPx

xripl.magnification.**tubeCenterPx**(*minimaIdxs*, *lineoutY=None*, *plotsFlag=False*)

Given the pixel positions of the tube edges, obtains the tube center in pixels for a tube target.

Parameters

- **minimaIdxs** (*tuple*) – A tuple of ints corresponding to the pixel positions of the two minima in intensity found in the radiograph by lineoutMinima(). These minima correspond to the inner edge of the target tube.
- **lineoutY** (*numpy.ndarray*) – Intensity values of the lineout from which tube edges were found. This is only used for plotting. Default is None.
- **plotsFlag** (*bool*) – Plots the lineout and overlays the pixel position of the tube edges and center. Default is False.

12.1.3 diameterPx

xripl.magnification.**diameterPx**(*minimaIdxs*)

Obtains the diameter of the tube in pixels when given the tube edges. Tube edges are obtained using lineoutMinima().

Parameters

minimaIdxs (*tuple*) – A tuple of ints corresponding to the pixel positions of the two minima in intensity found in the radiograph by lineoutMinima(). These minima correspond to the inner edge of the target tube.

12.1.4 magnificationUmPx

xripl.magnification.**magnificationUmPx**(*innerDiameterPx*, *stdDevPx=0*, *innerDiameterUm=500*, *innerDiameterErrUm=0*, *printFlag=False*)

Get magnification with propagated uncertainty in um/px when given target tube diameter in um, in px, and standard deviation on the diameter in px.

Parameters

- **innerDiameterPx** (*float*) – Measured inner diameter of the target tube in pixels.
- **stdDevPx** (*float*) – Standard deviation on the inner diameter in pixels. Default is zero, for no uncertainty.
- **innerDiameterUm** (*float*) – Known inner diameter of the target tube in micrometers. Default is set to 500 um, which was the nominal diameter for a Marble VC target.
- **innerDiameterErrUm** (*float*) – The error/uncertainty (one sigma) on the inner dimaeter of the target tube in micrometers. Default is 0 um.
- **printFlag** (*bool*) – Flag for printing the calculated magnification. Default is False.

12.1.5 magnificationAnalysis

xripl.magnification.**magnificationAnalysis**(*shot*, *camera*, *dataDir*, *px2Um*, *pxRef*, *umRef*, *xMin*, *xMax*, *yMin*, *yMax*, *cropDownstream=(1000, 1400)*, *innerDiameterUm=500*, *innerDiameterErrUm=0*, *saveDir=''*)

Get magnification from target radiograph by measuring inner tube wall of known diameter.

Parameters

- **shot** (*int*) – Shot number to read raw radiographic iamge data from.
- **camera** (*str*) – Framing camera for which radiographic image to read.
- **dataDir** (*str*) – Path where data for shot day is stored.
- **px2Um** (*float*) – Dummy value for magnification conversion to pass to initial image loading functions. This just sets the axes for those functions.
- **pxRef** (*tuple*) – Reference position in pixels. This can be a dummy value as it is only used by functions loading the image data.
- **umRef** (*tuple*) – Reference position in micrometers. This can be a dummy value as it is only used by functions loading the image data.
- **xMin** (*int*) – Lower bound pixel for cropping in x-direction.
- **xMax** (*int*) – Upper bound pixel for cropping in x-direction.
- **yMin** (*int*) – Lower bound pixel for cropping in y-direction.
- **yMax** (*int*) – Upper bound pixel for cropping in y-direction.
- **cropDownstream** (*tuple*) – Tuple of values defining the axial pixel bounds of the downstream end of the tube. This is where the tube diameter measurement in pixels will be taken. Default is (1000, 1400).
- **innerDiameterUm** (*float*) – The measured inner diameter of the target tube in micrometers. This is used the obtain the magnification in um/px. The default is 500 um, which is the nominal value of the Marble VC tube.
- **innerDiameterErrUm** (*float*) – The error/uncertainty (one sigma) on the inner dimaeter of the target tube in micrometers. This is used to get the uncertainty on the magnification. Default is 0 um.
- **saveDir** (*str*) – Directory for saving images and data. Default is empty string, which does not save the data.

CHAPTER

THIRTEEN

PLOT DEFAULTS (*XRIPL.PLTDEFAULTS*)

Created on Fri Oct 27 02:37:12 2017

Default plotting parameters

@author: Pawel M. Kozlowski

13.1 Functions

plot_line_shaded(xData, yData, yErrs[, label])	Generate a line plot with shaded region representing y-error bars.
plot_scatter_bars(xData, yData, yErrs[, label])	Generate a scatter plot with y-error bars.

13.1.1 plot_line_shaded

xripl.pltDefaults.**plot_line_shaded**(*xData*, *yData*, *yErrs*, *label=''*, ***kwargs*)

Generate a line plot with shaded region representing y-error bars. Can be run multiple times before plt.show(), to plot multiple data sets on the same axes. x axis data points y axis data points y axis errors

13.1.2 plot_scatter_bars

xripl.pltDefaults.**plot_scatter_bars**(*xData*, *yData*, *yErrs*, *label=''*, ***kwargs*)

Generate a scatter plot with y-error bars. Can be run multiple times before plt.show(), to plot multiple data sets on the same axes. x axis data points y axis data points y axis errors

CHAPTER

FOURTEEN

READER (*XRIPL.READER*)

Created on Mon Oct 22 16:42:16 2018

Reads radiographic images and create simple plots

@author: Pawel M. Kozlowski

14.1 Functions

openRadiograph(fileName)	Given the full filename (with path) to an hdf5 file containing radiographic data from an XRFC (x-ray framing camera) diagnostic on OMEGA, open the file and return the foreground and background image data as numpy arrays.
omegaDataSearch(dataDir[, camera])	Search given directory and subdirectories for HDF5 files of radiographs from OMEGA LLE XRFC data.
shotRadiograph(shotNum, camera, dataDir[, plots])	Opens a radiograph given a shot number, framing camera and directory containing shot data.

14.1.1 openRadiograph

xripl.reader.**openRadiograph**(*fileName*)

Given the full filename (with path) to an hdf5 file containing radiographic data from an XRFC (x-ray framing camera) diagnostic on OMEGA, open the file and return the foreground and background image data as numpy arrays.

fileName is a str

14.1.2 omegaDataSearch

xripl.reader.**omegaDataSearch**(*dataDir*, *camera='xrfc4'*)

Search given directory and subdirectories for HDF5 files of radiographs from OMEGA LLE XRFC data.

14.1.3 shotRadiograph

xripl.reader.**shotRadiograph**(*shotNum*, *camera*, *dataDir*, *plots=False*)

Opens a radiograph given a shot number, framing camera and directory containing shot data.

shotNum
[int] Omega shot number of radiograph to be opened.

camera
[str] Name of camera used on shotNum from which to open radiograph. For example, 'xrfc3' for x-ray framing camera 3.

dataDir
[str] Full path to directory containing shot data with radiographs.

plots
[bool] Flag for plotting the background and the foreground images. Default is False.

CHAPTER

FIFTEEN

SEGMENTATION (*XRIPL.SEGMENTATION*)

Created on Thu Jan 10 10:10:36 2019

Image segmentation tools for radiographic images.

@author: Pawel M. Kozlowski

15.1 Functions

merge_labels(labels_image, labels_to_merge, ...)	Utility function for merging different labeled regions.
nSegments(labels, n)	Get the n largest area segments from a segmented (labeled) image.
segmentContour(labels, segmentNumber[, plots])	Given a segmented image and a selected segment number, obtains the longest contour of that segment.
detectShock(image[, originalImage, ...])	This is a convenience function for applying watershed segmentation using regions of low gradient as markers, and using another gradient image for processing via watershed.

15.1.1 merge_labels

xripl.segmentation.**merge_labels**(*labels_image*, *labels_to_merge*, *label_after_merge*)

Utility function for merging different labeled regions.

Parameters

- **labels_image** (*numpy.ndarray*) – Labeled image.
- **labels_to_merge** (*numpy.ndarray*) – Array of label indices of labels to be merged.
- **label_after_merge** (*int*) – Label index used to overwrite labels in labels_to_merge.

15.1.2 nSegments

xripl.segmentation.**nSegments**(*labels*, *n*)

Get the n largest area segments from a segmented (labeled) image.

Parameters

- **labels** (*numpy.ndarray*) – A segmented 2D image.
- **n** (*int*) – Number of segments to select.

15.1.3 segmentContour

xripl.segmentation.**segmentContour**(*labels*, *segmentNumber*, *plots=False*)

Given a segmented image and a selected segment number, obtains the longest contour of that segment.

Parameters

- **labels** (*numpy.ndarray*) – A segmented 2D image.
- **segmentNumber** (*int*) – Which labeled segment to choose for obtaining contour.
- **plots** (*bool*) – Flag for plotting longest contour over binary image of segment.

15.1.4 detectShock

xripl.segmentation.**detectShock**(*image*, *originalImage=None*, *medianDisk=40*, *gradientDiskMarkers=10*, *gradientThreshold=10*, *gradientDiskWatershed=4*, *morphDisk=10*, *compactness=0*, *plots=True*, *nSegs=3*)

This is a convenience function for applying watershed segmentation using regions of low gradient as markers, and using another gradient image for processing via watershed.

Steps: 1st the image is denoised using a median filter 2nd the gradient of the image is obtained and thresholded to generate markers to be used for watershed segmentation. 3rd Morphological closing operation is applied to markers to make the region boundaries thicker and the marker regions more continous. 4th Another gradient image is generated, which will be processed via watershed. 5th The gradient image in step 4 and the markers from step 3 are processed using watershed segmentation.

Parameters

- **image** (*numpy.ndarray*) – Image to be processed for shock detection.
- **originalImage** (*numpy.ndarray*) – Original image, in case user want to plot contours and segments over a different image. Default is None.
- **medianDisk** (*int*) – Size of disk used in median filter application.
- **gradientDiskMarkers** (*int*) – Size of disk used in gradient function for determining watershed markers. This makes the gradient image smoother.
- **gradientThreshold** (*int*) – Maximum size of gradient to be used for selecting watershed markers.
- **gradientDiskWatershed** (*int*) – Size of disk used in gradient function for generating image to be processed by watershed. This makes the gradient image smoother.
- **morphDisk** (*int*) – Size of disk used in morphological closing of markers image. This cleans up the markers into more continuous regions. If zero, then this step is skipped.

- **compactness** (*float*) – Compactness parameter for watershed. See skimage.morphology.watershed().
- **plots** (*bool*) – Flag for plotting results
- **nSegs** (*int*) – Obtains contours of the largest segments by filled area.

CHAPTER

SIXTEEN

TUBE TRANSMISSION (*XRIPL.TUBETRANSMISSION*)

Created on Tue May 12 10:59:55 2020

Idealized estimate of what a 1D lineout radially through a cylindrical foam + tube would look like. This is to assist in correctly identifying which features in a radiograph correspond to which part of the target for spatial magnification calculation.

@author: Pawel M. Kozlowski

16.1 Functions

chord(radius, height)	Calculates the horizontal chord length through a circle at a given height.
tubeThickness(innerDiameter, outerDiameter, ...)	Calculates the chord lengths through the inner region (foam) and outer region (tube wall) for a given target.
transmission(density, opacity, length)	Calculates the transmission through a uniform material of given density, opacity, and length.
tubeTransmission(innerDiameter, ...)	Calculates the transmission through a tube with foam at a given height (radial coordinate) through the tube.
plotTubeTransmission(innerDiameter, ...[, plots])	Generates a plot transmission versus radial coordinate through a tube.
gaussianBlur(heightsArr, transmissions, ...)	heightsArr : numpy.ndarray

16.1.1 chord

xripl.tubeTransmission.**chord**(*radius*, *height*)

Calculates the horizontal chord length through a circle at a given height. If the height is larger than the circle's radius, then a chord length of zero is returned.

radius
[float] Radius of the circle. Must be in same units as height.

height
[float] Height along the circle at which to calculate the horizontal chord length.

16.1.2 tubeThickness

xripl.tubeTransmission.**tubeThickness**(*innerDiameter*, *outerDiameter*, *height*)

Calculates the chord lengths through the inner region (foam) and outer region (tube wall) for a given target. Here the tube is simply an anulus with the given inner and outer diameters as dimensions, and the region inside the tube is the "foam" region.

innerDiameter
[float] Inner diameter of the tube, which is also the diameter of the foam.

outerDiameter
[float] Outer diameter of the tube. Must be in same units as innerDiameter and height.

height
[float] Height at which to calculate the chord lengths through the tube and foam.

16.1.3 transmission

xripl.tubeTransmission.**transmission**(*density*, *opacity*, *length*)

Calculates the transmission through a uniform material of given density, opacity, and length.

density
[float] Mass density of the material in grams per cubic centimeter.

opacity
[float] Opacity of the material (also known as mass attenuation coefficinet) in squared centimeters per gram.

length
[float] Attenuation length through the material in centimeters.

16.1.4 tubeTransmission

xripl.tubeTransmission.**tubeTransmission**(*innerDiameter*, *outerDiameter*, *densityTube*, *densityFoam*, *opacityTube*, *opacityFoam*, *height*)

Calculates the transmission through a tube with foam at a given height (radial coordinate) through the tube.

innerDiameter
[float] Inner diameter of the tube, which is also the diameter of the foam. In units of centimeters.

outerDiameter
[float] Outer diameter of the tube. In units of centimeters.

densityTube
[float] Mass density of the tube material in grams per cubic centimeter.

densityFoam
[float] Mass density of the foam material in grams per cubic centimeter.

opacityTube
[float] Opacity of the tube material (aka mass attenuation coefficient) in squared centimeters per gram.

opacityFoam
[float] Opacity of the foam material (aka mass attenuation coefficient) in squared centimeters per gram.

height
[float] Height at which to calculate the chord lengths through the tube and foam. In units of centimeters.

16.1.5 plotTubeTransmission

xripl.tubeTransmission.**plotTubeTransmission**(*innerDiameter*, *outerDiameter*, *densityTube*, *densityFoam*, *opacityTube*, *opacityFoam*, *heightsArr*, *plots=True*)

Generates a plot transmission versus radial coordinate through a tube.

innerDiameter
[float] Inner diameter of the tube, which is also the diameter of the foam. In units of centimeters.

outerDiameter
[float] Outer diameter of the tube. In units of centimeters.

densityTube
[float] Mass density of the tube material in grams per cubic centimeter.

densityFoam
[float] Mass density of the foam material in grams per cubic centimeter.

opacityTube
[float] Opacity of the tube material (aka mass attenuation coefficient) in squared centimeters per gram.

opacityFoam
[float] Opacity of the foam material (aka mass attenuation coefficient) in squared centimeters per gram.

heightsArr
[numpy.ndarray] Array of heights along the tube radial axis at which to calculate the chord lengths through the tube and foam for getting transmission. These can be negative and positive. In units of centimeters.

plots
[bool] Flag for generating plots. Default is True.

16.1.6 gaussianBlur

xripl.tubeTransmission.**gaussianBlur**(*heightsArr*, *transmissions*, *stdDevCm*, *innerDiameter*, *outerDiameter*, *plots=False*)

heightsArr
[numpy.ndarray] Array of heights along the tube radial axis at which chord lengths through the tube and foam were used for getting transmission. These can be negative and positive. In units of centimeters. See plotTubeTransmission().

transmissions
[numpy.ndarray] Transmission values through the tube corresponding to heightsArr. See plotTubeTransmission().

stdDevCm
[float] Standard deviation of Gaussian (in centimeters) to be used for blurring the transmissions curve.

innerDiameter
[float] Inner diameter of the tube, which is also the diameter of the foam. In units of centimeters. Used to overlay tube location on plots.

outerDiameter
[float] Outer diameter of the tube. In units of centimeters. Used to overaly tube location on plots.

plots
[bool] Flag for plotting Gaussian used in convolution and blurred transmissions curve.

CHAPTER

SEVENTEEN

VISUALIZATIONS (*XRIPL.VISUALIZATIONS*)

Created on Mon Oct 22 16:44:47 2018

Convenience functions for visualizing radiographic data and processed data from XRIPL.

@author: Pawel M. Kozlowski

17.1 Functions

spatialConversion(img, px2Um, pxRef, umRef)	Generates spatial extents of image in micrometers, by using the image shape, pixel to micrometer conversion factor, and a reference position defined in both pixel and micrometer coordinates.
overlayTube(img, extent[, method])	Overlays known coordinates of tube based on shock tube dimensions onto the given image.
cropImgCalibrated(img, px2Um, pxRef, umRef, ...)	Crops the image and provides updated extents for spatial calibration of the image.
rotateImgCalibrated(img, px2Um, pxRef, umRef)	Rotates the image 90 degrees in counter clockwise direction, and provides updated extents for spatial calibration of the image.
lineoutsComparison1(imgRaw, imgDenoised, ...)	Plots normalized lineouts for raw, denoised, and cleaned iamges for comparison.
lineoutsComparison2(imgDenoised, imgFlat, px)	Compares pseudo-flatfielded lineout against denoised lineout.

17.1.1 spatialConversion

xripl.visualizations.**spatialConversion**(*img*, *px2Um*, *pxRef*, *umRef*, *method='vertical'*)

Generates spatial extents of image in micrometers, by using the image shape, pixel to micrometer conversion factor, and a reference position defined in both pixel and micrometer coordinates.

img
[numpy.ndarray] Image for getting spatial conversion coordinates.

px2Um
[float] Conversion factor micrometers / pixel for the image.

pxRef
[tuple] Reference position in pixels.

umRef
[tuple] Reference position in micrometers.

method
[str] Marble images are horizontal by default, but we make them vertical to match Ranjan SBI paper with shock propagating downward. Default shock direction is vertical.

17.1.2 overlayTube

xripl.visualizations.**overlayTube**(*img*, *extent*, *method='vertical'*)

Overlays known coordinates of tube based on shock tube dimensions onto the given image.

img
[numpy.ndarray] Spatially calibrated radiograph.

extent
[list] Extent coordinates for plotting the spatially calibrated image using plt.imshow(). See spatialCovnersion().

method
[str] Orientation of image. This should be the same as used in spatialConversion() to get extent.

17.1.3 cropImgCalibrated

xripl.visualizations.**cropImgCalibrated**(*img*, *px2Um*, *pxRef*, *umRef*, *xMin*, *xMax*, *yMin*, *yMax*, *method='horizontal'*, *plots=False*)

Crops the image and provides updated extents for spatial calibration of the image.

img
[numpy.ndarray] Image to be cropped.

px2Um
[float] Conversion factor micrometers / pixel for the image.

pxRef
[tuple] Reference position in pixels for the uncropped image. This is based on a known feature, such as a fiducial or filter edge.

umRef
[tuple] Reference position in micrometers. This is the same reference used for pxRef, but with the known target coordinates from metrology/design.

xMin
[int] Lower bound pixel for cropping in x-direction.

xMax
[int] Upper bound pixel for cropping in x-direction.

yMin
[int] Lower bound pixel for cropping in y-direction.

yMax
[int] Upper bound pixel for cropping in y-direction.

method
[str] Direction of shock propagation in the image. Default shock direction is horizontal.

plots
[bool] Flag for plotting cropped image with spatially calibrated axes. Default is False.

17.1.4 rotateImgCalibrated

xripl.visualizations.**rotateImgCalibrated**(*img*, *px2Um*, *pxRef*, *umRef*, *method='vertical'*, *plots=False*)

Rotates the image 90 degrees in counter clockwise direction, and provides updated extents for spatial calibration of the image.

img
[numpy.ndarray] Image to be rotated.

px2Um
[float] Conversion factor micrometers / pixel for the image.

pxRef
[tuple] Reference position in pixels for the unrotated image. This is based on a known feature, such as a fiducial or filter edge.

umRef
[tuple] Reference position in micrometers. This is the same reference used for pxRef, but with the known target coordinates from metrology/design.

method
[str] Direction of shock propagation in the newly rotated image. Default shock direction is vertical.

plots
[bool] Flag for plotting rotated image with spatially calibrated axes. Default is False.

17.1.5 lineoutsComparison1

xripl.visualizations.**lineoutsComparison1**(*imgRaw*, *imgDenoised*, *imgCleaned*, *px*, *extent=None*, *showFlag=True*)

Plots normalized lineouts for raw, denoised, and cleaned iamges for comparison. Plot shows how well noise is removed and how well features are preserved.

imgRaw
[numpy.ndarray] Raw radiographic image.

imgDenoised
[numpy.ndarray] Denoised (median filtered) radiographic image.

imgCleaned
[numpy.ndarray] Cleaned (morphologically filtered) radiographic image.

px
[int] Pixel position at which to take the lineouts.

extent
[list] Extents defining spatial axes. Used to form radial axis in um. Default is None, which falls back on radial axis in pixels.

showFlag
[bool] Flag for showing the plotted image. Set to False if you want to further modify the plot. Default is True.

17.1.6 lineoutsComparison2

xripl.visualizations.**lineoutsComparison2**(*imgDenoised*, *imgFlat*, *px*, *extent=None*, *showFlag=True*)

Compares pseudo-flatfielded lineout against denoised lineout. Pseudo-flatfielded lineout typically includes cleaning via morphological filtering, which can distort the image, whereas denoising is done through median filtering, which typically preserves features. This means this comparison is important for showing whether features are preserved through the morphological filtering step and whether the pseudo-flatifield suitably flattens the intensity curve compared to what we expect. In addition, this overlays expected tube positions for Marble VC 16A target.

imgDenoised
[numpy.ndarray] Denoised (median filtered) radiographic image.

imgFlat
[numpy.ndarray] Pseudo-flatfielded image (using over-blurring).

px
[int] Pixel position at which to take the lineouts.

extent
[list] Extents defining spatial axes. Used to form radial axis in um. Default is None, which falls back on radial axis in pixels.

showFlag
[bool] Flag for showing the plotted image. Set to False if you want to further modify the plot. Default is True.

CHAPTER

EIGHTEEN

INDICES AND TABLES

- genindex
- modindex
- search

PYTHON MODULE INDEX

X

xripl.clean, 13
xripl.contour, 15
xripl.contrast, 17
xripl.films, 19
xripl.filters, 21
xripl.instrument, 23
xripl.magnification, 25
xripl.pltDefaults, 29
xripl.reader, 31
xripl.segmentation, 33
xripl.tubeTransmission, 37
xripl.visualizations, 41

INDEX

B

boxFilter() (*in module xripl.filters*), 21

C

chord() (*in module xripl.tubeTransmission*), 37
cleanArtifacts() (*in module xripl.clean*), 13
cropImgCalibrated() (*in module xripl.visualizations*), 42

D

detectShock() (*in module xripl.segmentation*), 34
diameterPx() (*in module xripl.magnification*), 26

E

enhanceRadiograph() (*in module xripl.clean*), 14
equalize() (*in module xripl.contrast*), 17

F

flatten() (*in module xripl.clean*), 14

G

gaussianBlur() (*in module xripl.tubeTransmission*), 39

L

lineoutMinima() (*in module xripl.magnification*), 25
lineoutsComparison1() (*in module xripl.visualizations*), 43
lineoutsComparison2() (*in module xripl.visualizations*), 44

M

magnificationAnalysis() (*in module xripl.magnification*), 27
magnificationUmPx() (*in module xripl.magnification*), 26
merge_labels() (*in module xripl.segmentation*), 33
module
    xripl.clean, 13
    xripl.contour, 15
    xripl.contrast, 17
    xripl.films, 19
    xripl.filters, 21
    xripl.instrument, 23
    xripl.magnification, 25
    xripl.pltDefaults, 29
    xripl.reader, 31
    xripl.segmentation, 33
    xripl.tubeTransmission, 37
    xripl.visualizations, 41

N

nContours() (*in module xripl.contour*), 15
nSegments() (*in module xripl.segmentation*), 34

O

omegaDataSearch() (*in module xripl.reader*), 31
openFilm() (*in module xripl.films*), 19
openRadiograph() (*in module xripl.reader*), 31
openSyntheticRadiograph() (*in module xripl.instrument*), 23
overlayTube() (*in module xripl.visualizations*), 42

P

plot_line_shaded() (*in module xripl.pltDefaults*), 29
plot_scatter_bars() (*in module xripl.pltDefaults*), 29
plotTubeTransmission() (*in module xripl.tubeTransmission*), 39

R

rotateImgCalibrated() (*in module xripl.visualizations*), 43

S

saveHdf() (*in module xripl.instrument*), 23
segmentContour() (*in module xripl.segmentation*), 34
shotRadiograph() (*in module xripl.reader*), 32
spatialConversion() (*in module xripl.visualizations*), 41

T

transmission() (*in module xripl.tubeTransmission*), 38
tubeCenterPx() (*in module xripl.magnification*), 26

tubeThickness() (*in module xripl.tubeTransmission*), 38
tubeTransmission() (*in module xripl.tubeTransmission*), 38

X

xripl.clean
    module, 13
xripl.contour
    module, 15
xripl.contrast
    module, 17
xripl.films
    module, 19
xripl.filters
    module, 21
xripl.instrument
    module, 23
xripl.magnification
    module, 25
xripl.pltDefaults
    module, 29
xripl.reader
    module, 31
xripl.segmentation
    module, 33
xripl.tubeTransmission
    module, 37
xripl.visualizations
    module, 41