pvimage

Release 0.0.7

Ahmad Maroof Karimi, Benjamin Pierce, Justin Fada, Nicholas Pa

Apr 08, 2020

CONTENTS

CHAPTER

ONE

PROCCESSING FUNCTIONS

Created on Wed Aug 14 21:54:05 2019

@author: jlb269

process.**CellExtract**(*img*, *numCols*, *numRows*, *imgtype=''*)
Performs extraction of individual cells.

Finds lines in cell or module image and orients the image normal to the frame.

Args: img (numpy.ndarray): An image array numCols (int): number of cells across in module image numRows (int): number of cells down in module image imgtype (str): 'UVF' UV Fluorescence Image

'gradient' for unequal intensity across the image 'lowcon' low contrast between cell and background

or PL images without background subtraction

Returns: list of numpy.ndarrays

process.**GetLMError**(*x*, **args*)
Error function for detecting lens distortion using the image mask Called by autoLensCorrect function

process.**Mask**(*img*, *imgtype=''*)
Creates a mask of the cell area.

Thresholds the image to create a binary mask, with options for images with a gradient or low contrast.

Args: img (numpy.ndarray): An image array imgtype (str): 'UVF' UV Fluorescence Image

'gradient' for unequal intensity across the image 'lowcon' low contrast between cell and background

or PL images without background subtraction

Returns: numpy.ndarray

process.**PlanarIndex**(*img*, *imgtype=''*)
Performs re-orientation of module in frame.

Broadly, this function performs tasks of filtering, edge detection, corner detection, perspective transformation.

Args: img (numpy.ndarray): An image array

Returns: list of numpy.ndarray

process.**RotateImage**(*img*, *imgtype=''*)
Performs rotation of an image.

Finds lines in cell or module image and orients the image normal to the frame.

Args: img (numpy.ndarray): An image array imgtype (str): 'UVF' UV Fluorescence Image

'gradient' for unequal intensity across the image 'lowcon' low contrast between cell and background

or PL images without background subtraction

Returns: numpy.ndarray

process.**autoLensCorrect**(*mask*)

Obtains lens distortion correction parameters for an image mask

Uses the GetLMError function to find optimal n, f based on error of linear fitting on sides of convex hull. Best results on module images without darkened cells or dark cell edges. Recommended to find n,f on subset of images, then apply to a larger dataset.

Args: img (numpy.ndarray): input image array

Returns: n (float): barrel distortion factor f (float): lens focal length

process.**lensCorrect**(*img*, *n*, *f*)

Removes barrel or pincushion distortion from an image

Uses input parameters to remove lens distortion. Input parameters can be found with pipelines.GetLensCorrectParams function

Args: img (numpy.ndarray): input image array n (float): barrel distortion factor f (float): lens focal length

Returns: numpy.ndarray: array of the lens corrected image

CHAPTER

TWO

PIPELINE FUNCTIONS

Created on Thu Jan 30 10:29:52 2020

@author: jlbraid

pipelines.**FMpipeline**(*imagepath*, *savepath*, *n=None*, *f=None*, *numCols=None*, *numRows=None*, *savesmall=False*, *imgtype=''*)

Performs full-size module image processing steps, including lens correction, planar indexing, and cell extraction, if desired.

Lens correction is performed if n and f are provided. n and f can be found with the GetLensCorrectParams function. Cells are extracted if numCols and numRows are provided.

Args: imagepath (str): path to a raw image savepath (str): folder path for saving output numCols (int): number of cells across in module image numRows (int): number of cells down in module image savesmall (bool): Save a smaller .jpg version of the planar indexed

image with True. Default is False.

imgtype (str): 'UVF' UV Fluorescence Image 'gradient' for unequal intensity across the image 'lowcon' low contrast between cell and background

or PL images without background subtraction

Returns:

pipelines.**GetLensCorrectParams**(*imagepath*, *imgtype=''*)

Automatically detects lens correction parameters for an image using linear fitting of the module edges.

Args: imagepath (str): path to a raw image imgtype (str): 'UVF' UV Fluorescence Image

'gradient' for unequal intensity across the image 'lowcon' low contrast between cell and background

or PL images without background subtraction

Returns: n,f (float): lens correction parameters

pipelines.**MMpipeline**(*imagepath*, *savepath*, *numCols*, *numRows*, *stitch=False*, *imgtype=''*)

Performs minimodule image processing steps, including cell extraction, planar indexing, and recombination of cell images into a module image, if desired.

Args: imagepath (str): path to a raw image savepath (str): folder path for saving output numCols (int): number of cells across in module image numRows (int): number of cells down in module image stitch (bool): True if output image with cell-level images stitched

together is desired. Default is False

imgtype (str): 'UVF' UV Fluorescence Image 'gradient' for unequal intensity across the image 'low-con' low contrast between cell and background

or PL images without background subtraction

Returns:

CHAPTER

THREE

MINIMODULE PIPELINE

```
"""
@author: Jennifer Braid
"""
import pvimage as pvi
import glob2
from os import chdir
import os
import cv2
from matplotlib import pyplot as plt
wd=os.path.dirname(__file__)
chdir(wd)

ELfolder_path = '../data/Minimodules/*EL*'
ELfiles = glob2.glob(ELfolder_path)
save_path = '../data/out/'

#Extracting individual cells, saving each, and stitching into a combined image
for file in ELfiles:
    try:
        pvi.pipelines.MMpipeline(file,save_path,2,2,True)
    except OverflowError:
        pvi.pipelines.MMpipeline(file,save_path,2,2,True,'lowcon')

PLfile = '../data/Minimodules/MMPL.tiff'
pvi.pipelines.MMpipeline(PLfile,save_path,2,2,True,'lowcon')

saved = glob2.glob('../data/out/*')
for im in saved:
    plt.imshow(cv2.imread(im))
    plt.show()

#Demonstrating the pipeline
img = cv2.imread(PLfile)
plt.imshow(img)
plt.show()

mask = pvi.process.Mask(img, 'lowcon')
plt.imshow(mask)
plt.show()

cells = pvi.process.CellExtract(img, 2, 2)

for cell in cells:
    plt.imshow(cell)
```

(continues on next page)

(continued from previous page)

```
    plt.show()
    plt.imshow(pvi.process.Mask(cell,'lowcon'))
    plt.show()

planarindexed = []
for cell in cells:
    planarindexed.append(pvi.process.PlanarIndex(cell,'lowcon'))

for cell in planarindexed:
    plt.imshow(cell)
    plt.show()
```

CHAPTER

FOUR

FULL MODULE PIPELINE

```
"""
@author: jlbraid
"""
import pvimage as pvi
import glob2
from os import chdir
import os
import cv2
from matplotlib import pyplot as plt
wd=os.path.dirname(__file__)
chdir(wd)

folder_path = '../data/FullSizeModules/*'
files = glob2.glob(folder_path)
save_path = '../data/out/'

#Applying lens correction, planar indexing the module, and extracting cells
for file in files:
    n,f = pvi.pipelines.GetLensCorrectParams(file)
    #If a large batch of images, get average LC parameters for a subset of good
    #images, and apply to images with the same camera setup
    pvi.pipelines.FMpipeline(file,save_path,n,f,10,6,True)

#Demonstrating the pipeline
img = cv2.imread(files[0])
plt.imshow(img)
plt.show()

n,f = pvi.pipelines.GetLensCorrectParams(files[0])

lc = pvi.process.lensCorrect(img, n, f)
plt.imshow(lc)
plt.show()

mask = pvi.process.Mask(lc)
plt.imshow(mask)
plt.show()

pi = pvi.process.PlanarIndex(lc)
plt.imshow(pi)
plt.show()

cells = pvi.process.CellExtract(pi, 10, 6)
```

(continues on next page)

(continued from previous page)

```
for cell in cells:
    plt.imshow(cell)
    plt.show()
```

CHAPTER

FIVE

INDICES AND TABLES

- genindex
- search

CHAPTER

SIX

DESCRIPTION

PVimage is a Python3 package developed by the SDLE Research Center at Case Western Reserve University in Cleveland OH.

This work was supported by the U.S. Department of Energy's Office of Energy Efficiency and Renewable Energy (EERE) under Solar Energy Technologies Office (SETO) Agreement no. DE-EE-0008172. The work of Jennifer L. Braid was supported by the U.S. Department of Energy (DOE) Office of Energy Efficiency and Renewable Energy administered by the Oak Ridge Institute for Science and Education (ORISE) for the DOE. ORISE is managed by Oak Ridge Associated Universities (ORAU) under DOE Contract no. DE-SC0014664.

The process by which Electroluminescent images of photovoltaic modules are captured leads to variation in module orientation between images. To ensure the images are uniformly oriented and registered for analysis, we created an image processing pipeline, which has been discussed in our previous work [1,2,3,4]. Filtering and thresholding methods are used to initially preprocess the data to remove barrel distortion, reduce noise, and remove unimportant background data. With a noise-reduced image, a convex hull algorithm is used to identify cell areas and mark them as a "1" (white pixel) while every other pixel is assigned a "0" (black pixel). A series of 1-D x-axis and y-axis parallel slices are taken through the binary array to identify the steps up (0 to 1) and steps down (1 to 0) across the slice. These steps correspond to the module edge. A regression model is fit to the points along the module edge, and the intersections of the edge lines identify the corners of the PV module. A perspective transformation is, then, applied to uniformly orient and planarize the module image resulting in the final planar-indexed module image ready for subsequent analysis.

1. J.S. Fada, M.A. Hossain, J.L. Braid, S. Yang, T.J. Peshek, R.H. French, Electroluminescent Image Processing and Cell Degradation Type Classification via Computer Vision and Statistical Learning Methodologies, in: 2017 IEEE 44th Photovoltaic Specialist Conference (PVSC), 2017: pp. 3456–3461. https://doi.org/10.1109/PVSC.2017.8366291.

2. A.M. Karimi, J.S. Fada, J. Liu, J.L. Braid, M. Koyutürk, R.H. French, Feature Extraction, Supervised and Unsupervised Machine Learning Classification of PV Cell Electroluminescence Images, in: 2018 IEEE 7th World Conference on Photovoltaic Energy Conversion (WCPEC) (A Joint Conference of 45th IEEE PVSC, 28th PVSEC 34th EU PVSEC), 2018: pp. 0418–0424. https://doi.org/10.1109/PVSC.2018.8547739.

3. Ahmad M. Karimi, Justin S. Fada, Mohammad A. Hossain, S. Yang, Timothy J. Peshek, Jennifer L. Braid, Roger H. French, Automated Pipeline for Photovoltaic Module Electroluminescence Image Processing and Degradation Feature Classification, IEEE Journal of Photovoltaics. (2019) 1–12. https://doi.org/10.1109/JPHOTOV.2019.2920732.

4. Ahmad Maroof Karimi, Justin S. Fada, Nicholas A. Parrilla, Benjamin G. Pierce, Mehmet Koyutürk, Roger H. French, Jennifer L. Braid, Mechanistic Insights into Photovoltaic Module Performance: Application of Computer Vision and Machine Learning on Electroluminescence Images and Current-Voltage Characterization, IEEE Journal of Photovoltaics. (n.d.). https://doi.org/10.1109/JPHOTOV.2020.2973448.

PYTHON MODULE INDEX

p

INDEX