**OpenUVF**

Development Plan

**Project Overview**

OpenUVF will provide a set of tools to simplify the post-processing and analysis of UVF images. Principal features include: automatic segmentation of raw UVF images into modules and cells, automatic detection of cracks in UVF images using TensorFlow’s object\_detection API, a combined, simplified script to assist with custom training of further models, and a set of secondary functions/scripts to assist with training set development, UVF analysis, and model evaluation.

OpenUVF’s ultimate purpose is to demonstrate the potential of UVF as a high-throughput, straightforward, in-field inspection technique for PV modules. With the tools demonstrated in OpenUVF, it should become clear that UVF should be the go-to approach to inspecting PV plants above a certain age. The toolset should further provide potential licensors of Southern Company’s UVF technology a framework upon which to build. Accordingly, OpenUVF should be licensed permissively, such that our partners can expand our work and create/patent/license their own proprietary system(s) that allows them a competitive advantage in the field. The Apache License 2.0 would be an ideal license. Alternatively the Mozilla Public License 2.0 includes the protections and permissions necessary.

In the below table, brief summaries of current and potential scripts are provided:

|  |  |
| --- | --- |
| Script | Description |
| detection.ipynb | Public-facing automatic crack detection script |
| training.ipynb | Public-facing combined detector training script |
| segmentation.ipynb (\*) | Public-facing automatic module/cell segmentation script |
| evaluation.ipynb (\*) | Public-facing detection model evaluation script |
| detection\_utils.py | Collection of utility scripts for detection |
| training\_utils.py | Collection of utility scripts for training |
| evaluation\_utils.py (\*) | Collection of utility scripts for evaluation |
| refactor\_sets.ipynb | Utility script to anonymize image sets before release |
| map\_uvf.ipynb (\*) | Public-facing utility for correlating images and modules |
| setup.py (\*) | Utility script to automatically locate/download packages |
| segmentation.m | MATLAB automatic module/cell segmentation script |
| binary\_conversion.m | MATLAB module/cell binarization script |
| ellipse.m | MATLAB module segmentation helper script |
| find\_corners.m | MATLAB automatic module corner detection script |
| linear\_fit.m | MATLAB module segmentation helper script |
| InterX.m | MATLAB script that finds line intersections. EXTERNAL |
| manual\_classification.m | MATLAB manual crack classification script |
| uvf\_el\_comparison.m | MATLAB UVF/EL comparison/analysis tool script |

(\*) indicates a potential or under-development feature

**Function Outlines**

**Crack Detection and Classification**

1. detection.ipynb – Public facing Jupyter notebook that utilizes a pre-trained TensorFlow model to detect and classify cracks in UVF images.
   1. Features:
      1. Crack detection (fed Module or Cell Images)
      2. Crack statistics (Cell/Module/Plant Level)
      3. Crack classification (\*)
      4. Module assembly from cells
   2. Dependencies:
      1. detection\_utils.py
2. detection\_utils.py – Collection of utility scripts utilized for crack detection and classification as well as total plant statistics calculation.
   1. Functions:

**Model Development and Evaluation**

1. training.ipynb – Public-facing Jupyter notebook that manages the entire training pipeline process.
   1. Features:
      1. Set Refactoring (anonymize data set)
      2. Image Augmentation
      3. Conversion to TFRecords
      4. Automatic Pipeline Config Update
      5. Conducts Training
   2. Utilities:
2. training\_utils.py
3. evaluation.ipynb (\*)
4. evaluation\_utils.py (\*)
5. manual\_classification.m

**Image Segmentation**

1. *segment\_images.m (MATLAB) –*A function that segments UVF images into modules and cells. Returns color and binary versions of all full modules in the image and their corresponding cells.
   1. *Features:*
      1. *Module/Cell binarization*
      2. *Module corner detection*
      3. *Module perspective correction*
      4. *Cell segmentation*
   2. *Helper Functions:*
   3. *Dependencies:*
      1. MATLAB Image Processing Toolbox
      2. *InterX – Available at:* [*https://www.mathworks.com/matlabcentral/fileexchange/22441-curve-intersections*](https://www.mathworks.com/matlabcentral/fileexchange/22441-curve-intersections)

**Training Set Development**

Given the early stage of this utility, tools are necessary to assist with the development of a proper training set.

1. *manual\_classification.m (MATLAB)* 
   1. Description: A script that assists with the initial categorization of cells for training set development. Can be used for classification regardless of intent for training set development.
2. *train\_detector.ipynb (Python)*
   1. *Description:* Public facing jupyter notebook that combines the various elements
   2. *Features:*
      1. Image Augmentation – Increases the size training and evaluation datasets by applying consistent transforms that are sufficiently dissimilar to the base image.
      2. TFRecord Creation – Converts the training and evaluation datasets from images (PNGs) and annotations (xmls) to TensorFlow’s TFRecord format, which is necessary for running their models.
   3. *Helper Functions:*
   4. *Dependencies:*
      1. *tensorflow and tensorflow-gpu*
      2. *opencv*
      3. *PIL*
      4. *Augmentor*
      5. *lxml*

**Potential Features Outline**