AERIAL THERMOGRAPHY – DELIVERY

This service delivery involves data analysis of the defects in a solar plant site using the relevant **tiff images** and **kml files** using python automation techniques.

Folder name – highlighted in yellow color

File name – highlighted in blue color

Scripts – highlighted in grey color

The complete process involves in a AT delivery will be explained below :

INPUT FILES : (From GIS team) – GIS_TO_AUTOMATION

- Front_Page.JPG
- KML
  - CAD.kml
  - Inverter_Boundary.kml / Outer_Boundary.kml
  - Thermal_Defects.kml
- Fullgreyscale
  - Fullgreyscale.tif
- Fullinferno
  - Fullinferno.tif

The following points are to be done before starting the process for project delivery:

- Go to Start ---- Remote Desktop Connection ---- IP (106.51.3.224:3362) ---- Connect to any one of the logins

Username1 : deepanshu ; Password : qwertykey

(or)

Username2 : tamil ; Password : Tapl@2017

Preferred logins: Scripts 0-11 --- deepanshu login (has both python (env) and python (solar) --- interpreter settings)

Scripts 12-14 --- tamil login (has only python (solar) --- interpreter settings)

NOTE: Run Script 14 in tamil login since the process will be completed within few minutes

- Collect the STRING ID from GIS team

  Eg: 1=2X15 (2 --- no.of rows , 15 --- no.of columns) ; 2=2X30
- Make sure to have access to project summary sheet, defect calculation sheet and sample report document (to prepare project summary report -- after running 12th script)
- Capitalize each word (separated by underscore) in the project name

  Eg: Patua_Solar
- Stick to the following naming conventions for the files present inside the project folder :
  - GIS_TO_AUTOMATION/

    KML/

    CAD.kml

    Inverter_Boundary.kml

    Thermal_Defects.kml

    Fullgreyscale/

    Fullgreyscale.tif

    Fullinferno/

    Fullinferno.tif

- If necessary (with the help of find and replace “Ctrl+H”) make STRING_ID: instead of String ID: in CAD.kml and make Inverter No: instead of INVERTER_NO: in Inverter_Boundary.kml

➢ Apart from the GIS_TO_AUTOMATION present inside the project folder, following folders need to be copied inside the project folder :

1. Seagate Expansion Drive / Sample Folder / CAD
2. Seagate Expansion Drive / Sample Folder / Defects
3. Seagate Expansion Drive / Sample Folder / IT_DELIVERY
4. For the Scripts:(One folder to be choose according to the defects)
    - Seagate Expansion Drive / Sample Folder / Regular defects/Scripts: Contains regular ones like Junction Box, Single Cell Hotspot, etc.
    - Seagate Expansion Drive / Sample Folder / Older defects/Scripts: Contain defects like Bypass diodes, Open Circuits, etc.
    - Seagate Expansion Drive / Sample Folder / New defects/Scripts: Contains additional defects to regular ones like Broken Glass, Missing Module, etc.

    NOTE: All the scripts folders have all three temperatures included, i.e., minimum, maximum, and average.

NOTE: Total folders present in the project folder : 5 (GIS_TO_AUTOMATION, CAD, Defects, IT_DELIVERY, Scripts)

➢ Copy the CAD.kml from GIS_TO_AUTOMATION / KML and paste it inside the CAD folder.

➢ Open the Scripts folder in PyCharm.

➢ Set the interpreter settings (right bottom) :

- Scripts 0-11 --- python(env)
- Scripts 12-14 --- python(solar)

PROCESS for AT Delivery :

0.string_wise_split.py : The first process is to create a split cad from the provided cad table. The type of split in each table will be assigned to some string id and provided in the description. The split specified for each id will be provided by the GIS team. In this script we will extract the description from the cad table kml and extract tables that fall under each string id into a separate kml file with the id as the filename.

- Replace the proper project path in the script
- Add the string id properly as provided by the GIS team
  Eg: STRING ID provided : 1=2X15 ; 2=2X30 ;3=3X45 => 3 string id's
  Change : string=["1", "2", "3"]
- Run the script
- Check the output of this script in the CAD/split/CAD folder.
- Try opening the Fullgreyscale.tif and check whether overlaying the split files on top of it is proper.

module_splitter_scripts.py : Once the string split is done, using the STRING ID provided by the GIS team, the table is splitted into module wise split cad. Once the splitting is done the module polygons will also be numbered for each table in a matrix manner based on their row and column (i.e. 1,1 , 2,2) and added to the description as well. Now we have the split cad for all the strings, all split kml files are combined together with QGIS tool to get the final split cad. The merged files are then exported with features containing only the description.

- Replace the proper project path in the script
- Add the string id properly as provided by the GIS team
  Eg: STRING ID provided : 1=2X15 ; 2=2X30

Change as follows :

if table type == 1 :

rows=2

columns=15

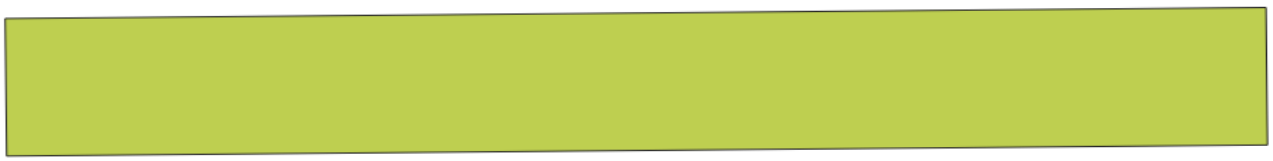
elif table type == 2 :

rows=2

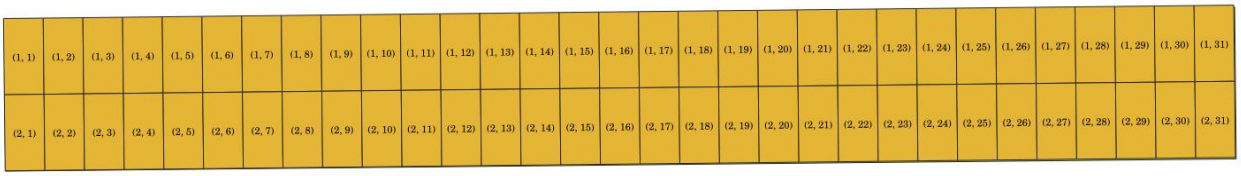
columns=30

- Run the script
- Check the output of this script in the CAD/output folder.
- Try opening the Fullgreyscale.tif and check whether overlaying the module split files on top of it is proper.
- To combine the split kml, all kml files present in CAD/output will be opened in QGIS.
- Go to Vector --- Data Management Tools --- Merge Vector Layers.
- In that window, select all the split cad files as input layer and run. It'll provide a temporary layer called merged with all tables.
- Now, right click on merged and select export and save feature as.
- In this window, provide output path "Seagate Expansion Drive/AERIAL_THERMOGRAPHY/<project folder>/CAD" and filename as "Final_CAD.kml" in filename and deselect all the fields to export except description and select OK.
- Check Final_CAD.kml file is generated in CAD folder

Input CAD table:



Output – Module level split CAD:



NOTE: If the placement of the solar panel is in slant manner, instead of module_splitter_scripts.py present in Scripts/module_scripts go for the module splitter files present in Seagate Expansion Drive/SampleFolder/Backup/yuvaraj Scripts/ Module_Splitter_Scripts (based on the no. of rows present in the STRING_ID provided : Eg: 1=2X15; 2=2X30 ; 3=2X60--- script to be used 2X.py)

1.inverter_wise_splitter.py: Using the inverter boundary provided, we will split the provided thermal defects kml into inverter wise kml files with the inverter name as the filename. The method used is polygon.contains() which will provide whether the polygon contains the object or not.

- Replace the proper project path in the script
- Run the script
- Check the output of this script in the Defects /INV_WISE_DEFECTS

2.INVERTER_wise_CAD_Spliter.py: Now the final split cad (CAD/ Final_CAD.kml) will be splitted into inverter wise kml files like the defects using the same method.

- Replace the proper project path in the script
- Run the script
- Check the output of this script in the Defects /INV_MOD_CAD

3. module_no_and_min_max_temprature.py: Using the inverter wise cad and defects kml we will extract the defect area for each defect in the given greyscale tiff (GIS_TO_AUTOMATION/Fullgreyscale/Fullgreyscale.tif) and extract the temperature values provided. We extract the minimum, maximum and average temperature in the defect module and add it to the defects description along with the module no. extracted from the respective cad as well. New kml files with temperature and module no will be created.

- Replace the proper project path in the script

  NOTE: From 25th to 40th line – Check the order of extracting with any one of the output file got from executing **1.inverter_wise_splitter.py (**Defects /INV_WISE_DEFECTS/). Comment the line if it is missing in the file. Most probably inverter no. will be missing in the file (if only one inverter present in the project). Thus try to hard code the inverter no. as "ITC_01" if necessary. Eg: inv_no = "ITC_01"

- Run the script
- Check the output of this script in the Defects / INV_WISE_DEFECTS_with_modno

4.Duplicates_Remover_mod_no.py: If the number of defects and inverters are high in number then there is a chance of duplicates present in the data. So, using this script will remove the duplicates by comparing the description of the defects. The files in Defects/INV_WISE_DEFECTS_with_modno folder will be replaced with files without duplicates.

- For most of the cases - skip this script.

5.Thermal_kml_split.py: From the inverter wise kml files with temperature data and module number, kml files for each defect type for each inverter will be extracted using the description. So, with inverter names as folder name and respective defect kml files with defect type as the filename will be created.

- Replace the proper project path in the script
- Run the script
- Check the output of this script in the Defects / DEFECTS_INV_CLASS_WISE
- Copy the files present in Defects /DEFECTS_INV_CLASS_WISE and paste it inside the IT_DELIVERY/kml folder.

6.descriptionChange.py: In the defect-wise kml files the descriptions are provided as normal text which will be converted into proper table data with html tags to display them in a table presentably. Along with adding the html tags we will also add the path for the defect image to display (AWS) will also be added.

- Replace the proper project path in the script
  NOTE: The project path should be checked carefully, since the naming convention in the path will directly affect in UI
- Run the script

7.Points_to_Polygons.py: Using the kml files from IT_DELIVERY/kml and inverter wise cad kml from Defects/INV_MOD_CAD the coordinates for the module polygon for each defect will be extracted and the description of the defects as well. Using them, new defect polygon kml's will be created with respective colour code for each defect. For defect such as Open String Tables and String Failure, the whole table will be defective so we will use the original

cad file from GIS_TO_AUTOMATION/KML/CAD.kml to get the coordinates for this defect kml.

- Replace the proper project path in the script
- Run the script
- Check the output of this script in the IT_DELIVERY/polykml

8.GLOBAL_KML_SIZE_MAKE_POLY.py: Using the inverter wise polygon kml, global kml file for each defect will be created. If the kml file size is large, then it will be created in multiple parts.

- Replace the proper project path in the script
- Run the script
- Check the output of this script in the IT_DELIVERY/global

9.GB_KML_GENERATOR.py : In the inverter wise boundary kml's (GIS_TO_AUTOMATION / KML/Inverter_Boundary.kml) we have the inverter names added in the description tag in respective inverter kml's. We should first merge all the inverter boundary kml's into one global boundary kml. Based on it we will hollow the boundary for the particular inverter considered and the other inverters we will give a fill.

- Replace the proper project path in the script
- Run the script
- Check the output of this script in the IT_DELIVERY/gb

10.Counting_json.py: Using the kml files from Defects /DEFECTS_INV_CLASS_WISE folder the number defects per inverter per defect will be extracted to generate inverter wise json and csv. Likewise, using the

global kml files from IT_DELIVERY/global a summary json and csv will be generated.

- Replace the proper project path in the script
- Run the script
- Check the output of this script in the IT_DELIVERY/counting

11.Excel_sheet_Gen.py: Using the Defects /DEFECTS_INV_CLASS_WISE kml files a excel report file will be generated with required details of all defects such as defect type, inverter no.(block no.), table no., module no., defect type, temperature (minimum, maximum, average temperature) and coordinates (latitude and longitude).

- Replace the proper project path in the script
- Run the script
- Check the output of this script in the IT_DELIVERY/report/Report.xlsx
- Cross check the total defect count with the total defect count given by GIS team.

12.Report_autommation.py: The excel file created will be converted into a pdf file using report lab library functions.

NOTE: Make sure to change the interpreter settings from env to solar before running the script

- Replace the proper project path in the script
- Run the script
- Check the output of this script in the IT_DELIVERY/report/merge/2.pdf
- Project Summary Report needs to be made ready before running the 13th script.

DOCUMENT PART : Project Summary Report

- Make a copy of the sample report document and change the title of the document as the Project name.
- With the help of Project summary sheet :

NOTE: While changing, make sure that the changes are affected wherever needed (i.e. Find and replace all – Ctrl+H with match case box checked in)

- In the 1st page of the document : change the location, plant size, date and the image

  NOTE: image is available in GIS_TO_AUTOMATION/ Front_Page.JPG

- In the 5th page : cross check with project summary sheet and change the acquisition summary.
- Copy the files from IT_DELIVERY/counting and paste it in the local system.
- Go to split files in CAD/output and open it in QGIS. For each split that is visible in the left bar of QGIS :
  - right click that split and select properties.
  - Change from no labels to single labels.
  - Set the value: Description Tag
  - Font : Century Schoolbook L
  - Apply --- OK
  - Zoom into one layer (If necessary reduce the font size) and take picture of that layer
  - Copy and paste it in the 8th page of the document.
- Open the summary.json (format it using Alt+Shift+F).
- In the defect calculation sheet, properly enter the values that are present in summary.json.

- Cross check each defect count and total defect count with the count sent by GIS team.
- Copy the values from the defect calculation sheet and paste it in the defect table present in the 6th page of the document.

➢ Share the doc by giving access to "anyone with the link" and copy that link. Paste the link in the remote desktop and download it as pdf

➢ Rename the pdf as "1.pdf" and copy the pdf and paste it in IT_DELIVERY/report/merge folder

NOTE : Make sure to restrict the access of the document once the above process is done.

13.Reports_merge.py: The report pdf file (Summary Report) will be named as 1.pdf and the excel pdf file will be named as 2.pdf. With this script the files will be merged as a single pdf file using pypdf library.

➢ Replace the proper project path in the script

➢ Run the script

➢ Check the output of this script in the IT_DELIVERY/report /Report_Final.pdf

➢ Compress the IT_DELIVERY/report/Report.xlsx and IT_DELIVERY/report /Report_Final.pdf , and name the zipped file as Report.zip (IT_DELIVERY/report)

14.ThermalCroping_lat.py: Now the defect images will be extracted from the GIS_TO_AUTOMATION/Fullinferno/Fullinferno.tif for all the defects by the following process. Using the Defects /INV_WISE_DEFECTS kml files, the

defect coordinates will be extracted. The coordinates will be converted to pixel coordinates using geotrans extracted from Fullinferno.tif . Using the pixel coordinates of a defect point a buffer region will be created to extract the defect module along with its nearby modules on each side. With the buffer region the image from inferno will be cropped and saved in the inverter folder in IT_DELIVERY/images

- Replace the proper project path in the script
- Run the script
- Check the output of this script in the IT_DELIVERY/images

THINGS TO BE DONE BEFORE UPLOADING THE FILES INTO AWS:

- Copy all the folders present in IT_DELIVERY/gb folder. Paste it in IT_DELIVERY/polykml (Merge the files present in gb folder with the existing folders in polykml).
- Rename the folders present inside IT_DELIVERY/polykml as INVERTER01,INVERTER02,…etc. (Eg: If the folder name is INV_01, then after renaming INVERTER01).
- Rename the global folder present in IT_DELIVERY as GLOBAL (i.e. IT_DELIVERY/global to IT_DELIVERY/GLOBAL).
- Copy the IT_DELIVERY/GLOBAL folder and paste it inside IT_DELIVERY/polykml.
- Rename the kml folder present in IT_DELIVERY as pointkml (i.e. IT_DELIVERY/kml to IT_DELIVERY/pointkml).
- Rename the polykml folder present in IT_DELIVERY as kml (i.e. IT_DELIVERY/polykml to IT_DELIVERY/kml).

---

3 THINGS TO BE UPLOADED ---- AWS:

- Report.zip file (IT_DELIVERY/report/Report.zip)
- images folder (IT_DELIVERY/images)
- kml folder (IT_DELIVERY/kml)

AWS:

- Make sure to have credentials to access AWS
- Login into AWS using the credentials
- Go to Services/S3/Buckets/datasee_ai_public_assets/2k23/AT/
- Go to the project folder if already exists or else create one folder named as project name.
- Inside the project folder upload the 3 things that are mentioned above.
- After the uploading process gets completed , click on the close button highlighted in green navigation bar.
- Now select the 3 things that are uploaded and go to Actions and click Make public using ACL.
- Once it gets over, close the AWS

- Copy the 4 files (inverter.json, inverter.csv, summary.json, summary.csv) present in IT_DELIVERY/counting and paste in to the local system.
- Copy Report.zip file present in IT_DELIVERY/report and paste in to the local system.

FILES TO BE SENT IN Project Delivery Channel (SLACK)

(Addressing @Adhityan T and @Karthikeyan N)

- inverter.json
- inverter.csv
- summary.json
- summary.csv
- Report.zip
