Short User Manual

General

This document serves to operate the program and not to explain the individual process steps. The detailed functional principle of the classification is described in the master thesis by Fabian Fabricius [1].

The available program is designed to run on a Linux operating system. Another variant can be executed on Windows. The program is programmed with Spyder 4.1.4. Therefore it is recommended to have at least this version installed. By opening the program as a project, the connections between the individual modules are activated and the descriptions can be read directly in the source code.

It is recommended to perform the evaluation using single images. The function to cut samples from a multiple image is implemented, but is not described further in this document.

For the first operation it is advisable to work through this document with the given data in order to understand the workflows.

Folder structure

In order for Python to find the images of the individual micrographs, the directory structure shown below must be followed. Non-compliance will lead to errors. In the "source" directory there is a folder with the necessary directory structure, which can serve as a template for analyses.

Steps:

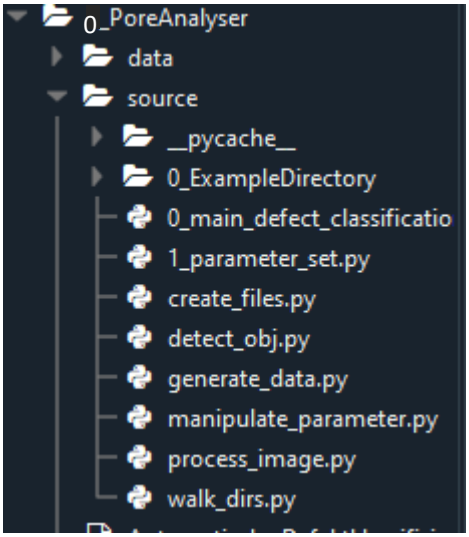
1. Open the "Home/sample analyses/0_PoreAnalyser" directoryKopiere den Ordner „0_ExampleDirectory“ aus dem „source“-Verzeichnis in das „data“-Verzeichnis
2. Copy the folder "0_ExampleDirectory" from the "source" directory into the "data" directory
3. Create folders for each analysis and name them i.e. "V1", "V2" and "V10 (according to your experiment ID)
4. Under "Home/sample analyses/1_Examples" there are sample images, which are now copied into the corresponding "1_Sample"-folder in the "data"-directory
5. After the steps have been performed, the basic framework for the analysis ready

1. Opening the Python project

1. Ppen a terminal in Linux
2. Open the Anaconda navigator with the command "anaconda-navigator“
3. Open Spyder from the Anaconda-Navigator
4. Close all open scripts if necessary
5. Open the project "0_PoreAnalyser" via the tab Projects -> Open Project... -> Home/sample analyses/0_PoreAnalyser -> Choose
6. If no folder structure is shown on the left side, active the project view via Projects -> Project (Ctrl + Shift + P)

2. Open the script

DDouble-click on the file "0_main_defect_classification.py" in the project tab to open the executable script in the editor.



Under the section #____Initialization BEGIN____# some specifications must be made in the editor, so that the test folders can be assigned correctly. The following table provides an overview of the necessary specifications.

The images of the experiments "V1" and "V10" were taken with a magnification of 20x, whereas "V2" was taken with a magnification of 10x. The dimensions of the micrographs are as follows:

V1 -> rectangular (4x2.5);

V2 -> circular (4x4);

V10 -> rectangular (4x5)

image_type	Specify whether several single images are to be generated from one image ("array") or whether single images are to be evaluated directly ("samples").	„samples“
magnification	Specification of the magnification of the microscope for each individual folder, in the order in which the folders are listed. [1]	[20,20,10]
sample_size	Specification of the dimensions of the specimens in the micrograph. The theoretical dimensions of the models are given here and not the measured ones. Specification in millimeters [1]	[(4,2.5), (4,5), (4,4)]
crop_factor	If a sample has very poor contours, the image is cropped by the amount entered from all sides. Default setting is 10%	0.1
circularity	Specification of how far gas pores may deviate from a perfect circle. Default setting is 95%	0.95
crack_ratio	Whether a pore is classified as a crack is indicated by the length-to-width ratio. Default setting is 5	5
print_defects	If True, each pore will be saved as a single image under the corresponding folder in 3_Pores.	True
print_proc	If "True", each individual sample is output under the Images tab during image processing. This allows the overview if all samples are cut properly.	False

[1] Die Ordner werden entsprechend ihrer Bezeichnung alphabetisch sortiert. Da bedeutet das die Ordner V1, V2 und V10 nicht in der gegebenen Reihenfolge bearbeitet werden. Python sortiert die Order folgendermaßen -> V1, V10, V2. Um zu überprüfen, welche Reihenfolge Python zugrunde liegt, kann die Variable „dir_list“ im Variablenmanager geöffnet werden, nachdem das Skript einmal ausgeführt wurde. Dort befinden sich für jeden Ordner ein Eintrag und die Reihenfolge innerhalb der Variable „dir_list“ ist auch die Reihenfolge mit der die Variablen in der oben genannten Tabelle angegeben werden müssen.

	This process needs a lot of CPU and therefore the evaluation takes longer.	
assign_data	If "True", the process parameters located in a JSON file in the 1_Samples folder are assigned to the samples. See section -> Process parameter file	True

With this information, the script is executable, but we have set the variable "assign_data" to True. This means that the program expects a JSON file where the associated process parameters can be found. So in the following step a JSON file has to be created.

4 Creation of the process parameter file

Double-click on the file "1_parameter_set.py" in the project tab to open the executable script in the editor. For each folder in the "data" directory, a JSON file must now be created. To do this, some experiment-specific information must be entered in the editor. For the three experiment folders this looks like this:

Variable	**V1**	**V2**	**V10**
name	"V1"	"V2"	"V10"
material	"FeCrCoNi"	"316L"	"316L"
supply_factor	3	3	3
geometry	[4,4,2.5]	[4,4,15]	[4,4,5]
scan_strategy	"90°"	"90°"	"90°"
spot_size	90	80	90
h_s *(Hatch)*	0.07	0.09	0.07
l_z *(layer thickness)*	0.05	0.05	0.05
P_l *(laser power)*	[100,250,300,150,150]	[160,175,190,205]	[350,350,300,150,150]
v_s *(markspeed)*	[200,300,300,400,600]	[450,450,800,450]	[200,400,200,400,600]
samples	["1","2","3","4","5"]	["0_0","0_1","0_2","0_3"	["01","02","03","04","05"]

Steps:

1. opening the script "1_parameter_set.py
2. input of the test specific variables
3. execute the script (green arrow)
4. python creates a JSON file in the "Home/sample analyses/0_PoreAnalyser/source" directory
5. move the JSON file to the corresponding experiment directory in the folder "1_Sample".

After the steps have been performed for each trial folder, each directory in the "data" folder contains the images to be evaluated and the corresponding JSON file in the "1_Sample" folder.

If no process parameters are to be linked to the images, the variable "assign_data" must be set to False in the "0_main_defect_classification.py" script.

5 Starting the analyes

After the described actions have been performed, the script "0_main_defect_classification.py" can be started by pressing the green arrow. Now the script cuts each image for each folder and analyzes the pores.

After the script has finished the process, the entire folders must be removed from "data" to make room for new test series.

All steps at a glance:

1. Create the folder structure.
2. Open the script "0_main_defect_classification.py" and enter the variables
    a. If process parameter should be assigned -> assign_data = True
    b. Create the process parameter files
3. Execute the script and wait for evaluation
4. After evaluation, remove the entire folders from the "data" directory and copy them to any location.

The evaluation of all images takes about 16 minutes in this case. So don't be surprised that it takes a little longer in some cases.

Literature:

[1] F. Fabricius, "Charakterisierung von Prozessfenstern in der additiven Fertigung durch computergestützte Klassifizierung von Materialdefekten," M. Sc. , Leibniz-Institut für Werkstofforientierte Technologien - IWT, Universität Bremen, Bremen, 2021.