USER'S MANUAL OF UNetGE

Chengdu University of Technology, Chengdu, China

January, 2022

USER'S MANUAL TABLE OF CONTENTS

1. GENERAL INFORMATION

- UNetGE is just windows version.
- UNetGE is developed by Ling Zeng and Lei Chen, and it is supported by the Natural Foundation of China (No. 42002294).
- There is a corresponding paper for UNetGE that illustrate the underlying principles to run this software: "UNetGE: a U-Net-based software at automatic grain extraction for image analysis of the grain size and shape characteristics".

2. INSTALLATION

- After the user obtain the installation file of UNetGE, please decompress it (circled in red in Fig. 1), then go into "main" file folder (in blue in Fig.1), and finally click "main.exe" (in pink in Fig. 1). It is to open the interface of UNetGE (in green in Fig. 1).

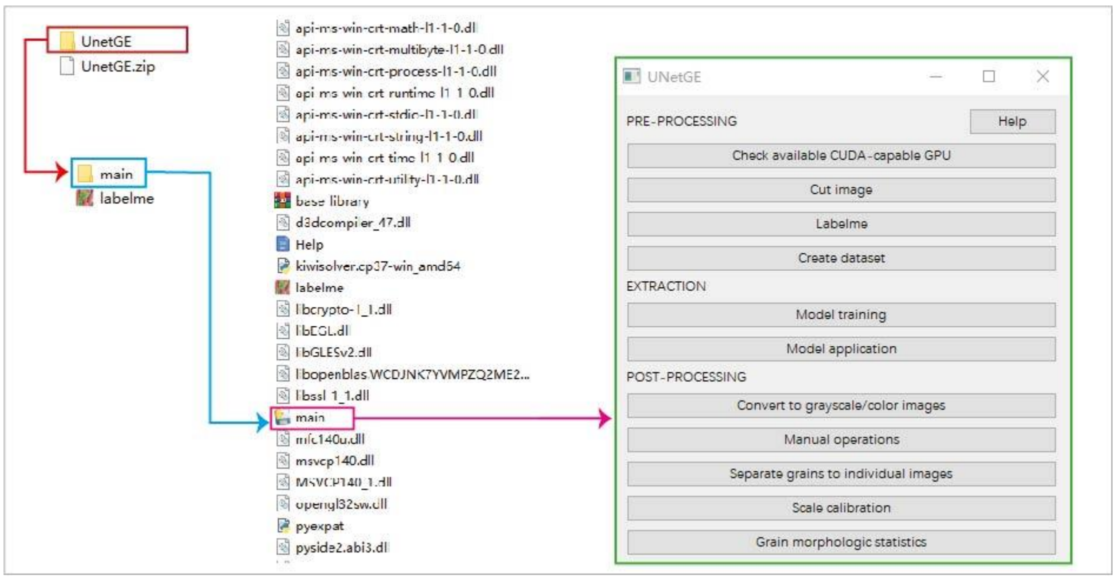

UnetGE
UnetGE.zip
main
labelme
api-ms-win-crt-math-l1-1-0.dll
api-ms-win-crt-multibyte-l1-1-0.dll
api-ms-win-crt-process-l1-1-0.dll
api-ms-win-crt-runtime-l1-1-0.dll
api-ms-win-crt-stdio-l1-1-0.dll
api-ms-win-crt-string-l1-1-0.dll
api-ms-win-crt-time-l1-1-0.dll
api-ms-win-crt-utility-l1-1-0.dll
base library
d3dcompiler_47.dll
Help
kiwisolver.cp37-win_amd64
labelme
libcrypto-1_1.dll
libEGL.dll
libGLESv2.dll
libopenblas.WCDJNK7YVMPZQ2ME2...
libssl-1_1.dll
main
mfc140u.dll
msvcp140.dll
MSVCP140_1.dll
opengl32sw.dll
pyexpat
pyside2.abi3.dll
UNetGE
PRE-PROCESSING
Help
Check available CUDA-capable GPU
Cut image
Labelme
Create dataset
EXTRACTION
Model training
Model application
POST-PROCESSING
Convert to grayscale/color images
Manual operations
Separate grains to individual images
Scale calibration
Grain morphologic statistics

Fig.1 the installation file of UNetGE and its interface

3. PRE-PROCESSING

- **“Check available CUDA-capable GPU” utility**

  Click the button of this utility, then the dialog will require to check whether the CUDA-capable GPU running environment is available or not. If the note dialog shows the GPU is not available, or you will use CPU for calculation or please install a NIVIDIA graphics card and CUDA so as that GPU is available. CPU calculation will possibly make running rather slowly. Moreover, the user can also use GPU Cloud Computing of e.g. Tencent cloud in the steps of Model training and Model application.

- **“Cut image” utility**

  Its purpose is cut large-size image into small-size pieces in order to facilitate labelling grains. Please select a folder with the files of *.jpg”, or ”*.jpeg “, or ”*png” as input, and create a folder to save the resultant pieces of images.

- **“Labelme” utility**

  This button is a link to a project created by the MIT computer science and artificial intelligence laboratory that is to prepare the training data.

  - In the interface of Labelme (Fig. 2a), “Open” is to just open one image, and “Open Dir” can simultaneously open all images in a folder.
  - When you Open a folder of images, you may use “Next Image” or “Prev Image” to choose one image to work on in the view window (Fig. 2a).
  - Please click “Create Polygons” (Fig. 2a) to start labelling the outlines of grains one

by one. When one enclosed polygon is finished for the outline of a grain, there is a dialog coming out (Fig. 2b) to ask the user to write down the class ID of this object (e.g., ID “plg” in Fig.2 denotes the category of plagioclase). Generally, this software can just extract one c   ategory of grains in each processing. Therefore, keep the same ID for all labeled grains in an image.

- Click “Edit Polygons”, the user can select one grain and modify its outline or delete its outline by click “Delete Polygons” (Fig. 2a). Another way is to use the “Polygon Labels” subpanel (the right side in Fig. 2a) to re-edit or delete the outlines of those labeled grains. The user can also use Key up and Key down of the keyboard to quickly check all labels in “Polygon Labels”.
- After labelling all grains of interest in a piece of image, please “Save” it as a json file (Fig. 2a). Then the user can select another piece of image for labelling.

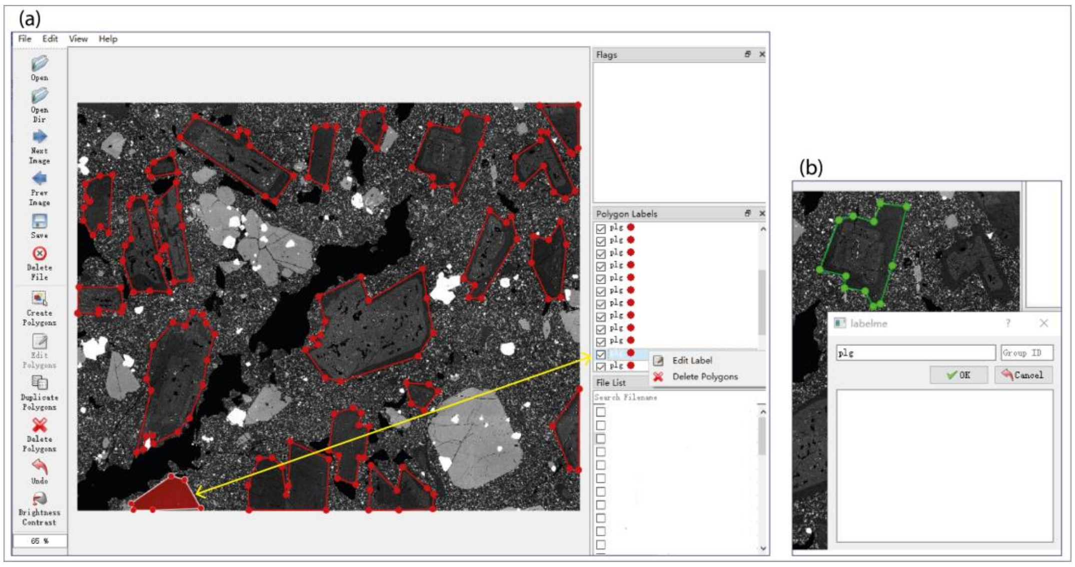
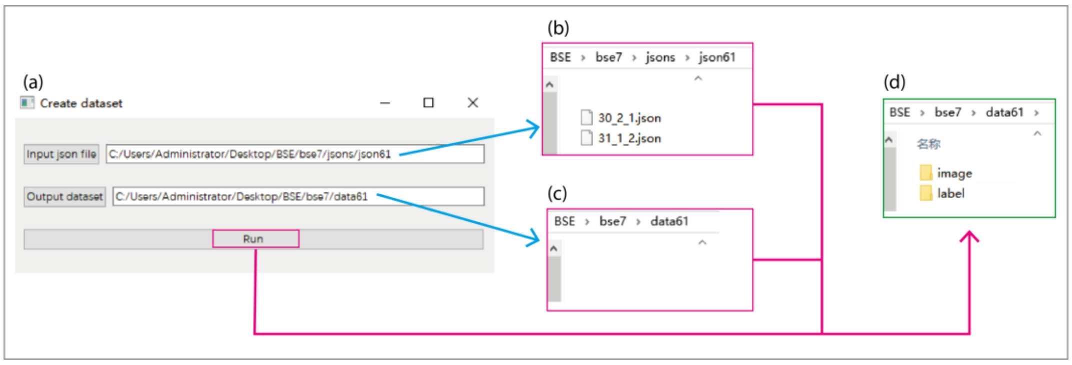


Fig. 2 The interface and view windows of Labelme

- **“Create dataset” utility**

Put the json files intended to create a dataset into a folder (Fig. 3b), and create a folder (Fig. 3c) for saving the dataset. Open the interface of this utility and click “Run” (Fig. 3a) and the result is shown in Fig. 3d.


Create dataset

Input json file: C:/Users/Administrator/Desktop/BSE/bse7/jsons/json61

Output dataset: C:/Users/Administrator/Desktop/BSE/bse7/data61

Fig. 3 How to create a dataset by json files

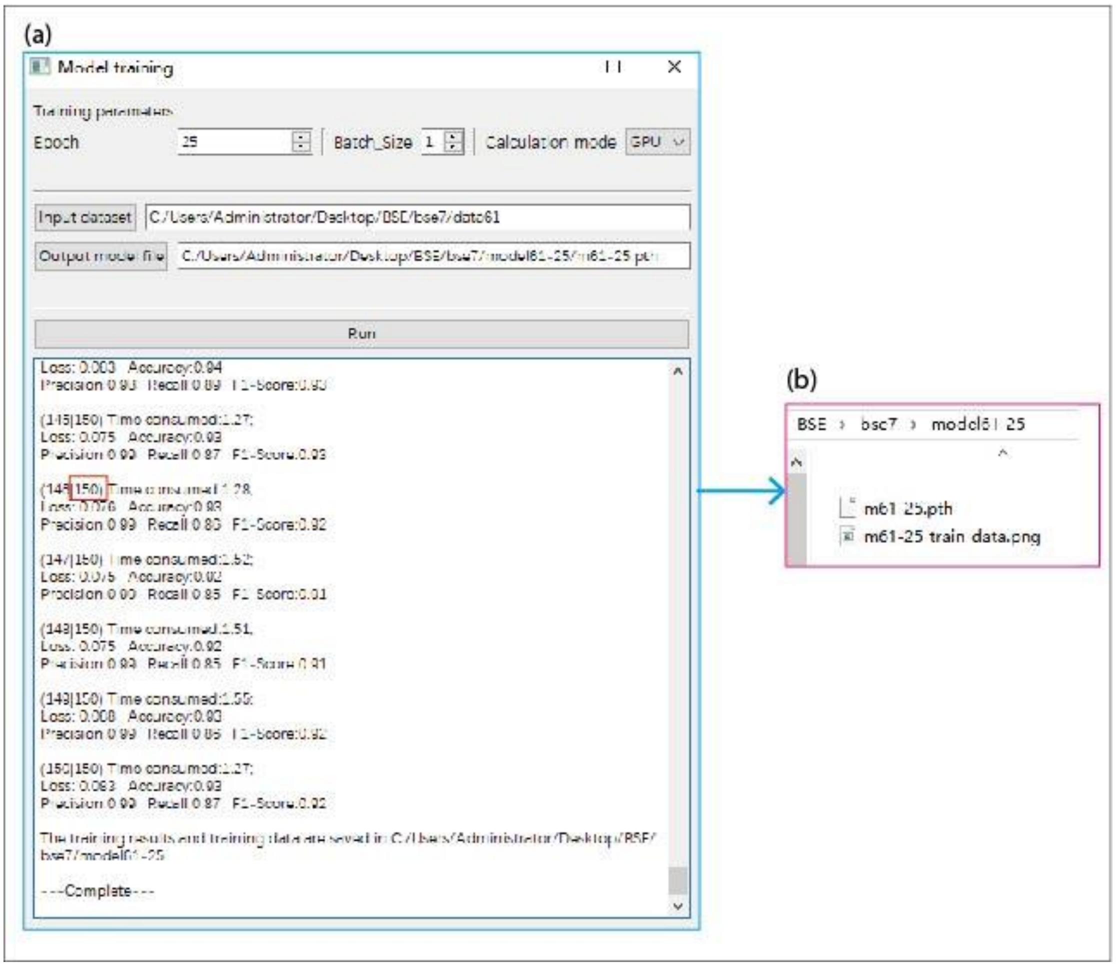
4. EXTRACTION

- **“Model training” utility**

Three parameters are needed filling out. In our applications, we used an eight-core 32GB GPU during the whole processing, and correspondingly we filled the three parameters as shown in Fig. 4: Epoch of 25, Batch_Size of 1, and Calculation mode of GPU. This may be a reference for the users’ applications.

It is noteworthy that CPU calculation will possibly much rather slower than GPU calculation, so please using GPU for calculation as possible. The input is a dataset created in the last step as shown in Fig. 3d, and the output is a “*.pth” file. Please name the model file before training, and the resultant model file is as Fig. 4b.

Furthermore, it is also noteworthy that the number delineated in red in Fig. 4a. If the video memory is not enough to train for the size of images, the automatic cutting function of UNetGE will cut images into smaller-size pieces to facilitate training. Therefore, the number in red in Fig. 4a denotes the total times of trainings after automatic cutting. And the bigger the number is, the smaller the size of those cutting pieces is and the longer the whole training will last. Approximately, if that number beyond one thousand, it would cost more than ten mins to running this model training.



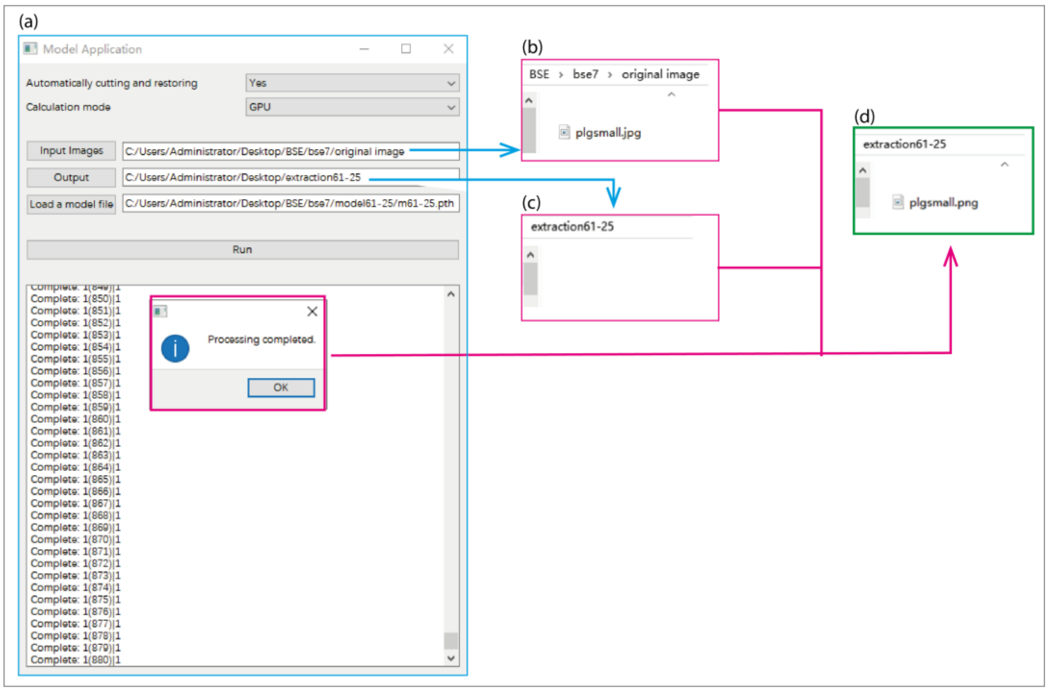
```
Loss: 0.083   Accuracy:0.94
Precision 0.93   Recall 0.89   F1-Score:0.93

(145|150) Time consumed:1.27;
Loss: 0.075   Accuracy:0.93
Precision 0.99   Recall 0.87   F1-Score:0.93

(146|150) Time consumed:1.28;
Loss: 0.076   Accuracy:0.93
Precision 0.99   Recall 0.86   F1-Score:0.92

(147|150) Time consumed:1.52;
Loss: 0.075   Accuracy:0.92
Precision 0.99   Recall 0.85   F1-Score:0.91

(148|150) Time consumed:1.51;
Loss: 0.075   Accuracy:0.92
Precision 0.99   Recall 0.85   F1-Score:0.91

(149|150) Time consumed:1.55;
Loss: 0.008   Accuracy:0.93
Precision 0.99   Recall 0.86   F1-Score:0.92

(150|150) Time consumed:1.27;
Loss: 0.083   Accuracy:0.93
Precision 0.99   Recall 0.87   F1-Score:0.92

The training results and training data are saved in C:/Users/Administrator/Desktop/BSE/
bse7/model61-25

---Complete---
```

Fig. 4 The interface and view window of model training and the resultant model file

- **"Model Application" utility**

  ➢ **"Automatically cutting and restoring"** is aimed at those images that are about to be applied to extract grains. If the sizes of the images are too big, it will possibly lead to processing rather slowly. This function will help to automatically cut big-size images into smaller pieces so as to facilitate with quick extraction and then restore to the original image. Therefore, it is better to choose “Yes” here (Fig. 5a).

- **“Calculation mode”** is or using GPU or using CPU for calculation.
- **“Input images”** is a folder containing all images intended for extraction application (e.g., Fig. 5b).
- **“Load a model file”** is to choose a trained model file (e.g., m61-25.pth in Fig. 4b).
- **“Output”** is a new folder created by the user to save the resultant images of grains extraction (Fig. 5c). When the processing of application is finished, the extraction result is saved in this folder (Fig. 5d). Note that the names of the outputted images are the same with those of their corresponding original images, and the only difference is the names of their file folders.
- It is noteworthy that no matter the original image is grayscale or color or binary, the resultant image of grains extraction in model application is just binary.



Automatically cutting and restoring: Yes
Calculation mode: GPU

Input Images: C:/Users/Administrator/Desktop/BSE/bse7/original image
Output: C:/Users/Administrator/Desktop/extraction61-25
Load a model file: C:/Users/Administrator/Desktop/BSE/bse7/model61-25/m61-25.pth

Run

```
Complete: 1(849)|1
Complete: 1(850)|1
Complete: 1(851)|1
Complete: 1(852)|1
Complete: 1(853)|1
Complete: 1(854)|1
Complete: 1(855)|1
Complete: 1(856)|1
Complete: 1(857)|1
Complete: 1(858)|1
Complete: 1(859)|1
Complete: 1(860)|1
Complete: 1(861)|1
Complete: 1(862)|1
Complete: 1(863)|1
Complete: 1(864)|1
Complete: 1(865)|1
Complete: 1(866)|1
Complete: 1(867)|1
Complete: 1(868)|1
Complete: 1(869)|1
Complete: 1(870)|1
Complete: 1(871)|1
Complete: 1(872)|1
Complete: 1(873)|1
Complete: 1(874)|1
Complete: 1(875)|1
Complete: 1(876)|1
Complete: 1(877)|1
Complete: 1(878)|1
Complete: 1(879)|1
Complete: 1(880)|1
```

Processing completed.

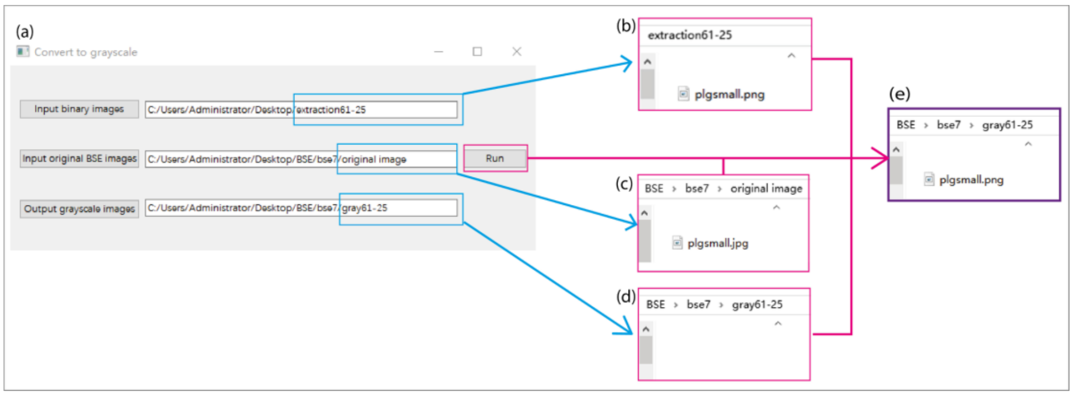
OK

Fig 5 How to finish model application for grains extraction

5. POST-PROCESSING

- **"Convert to grayscale/color images" utility**

Select the folder consisting of the resultant binary images of grains extraction in the step of Model Application (Fig. 6b or Fig. 5d) as the input. Create a folder for the output (Fig. 6d), and choose the original gray/color image (Fig. 6c). Then click "Run", the resultant images in gray or color will be produced (Fig. 6e). Note that the names of all images should be the same, the sole difference is that the names of folders to save them is varied.



Input binary images: C:/Users/Administrator/Desktop/extraction61-25

Input original BSE images: C:/Users/Administrator/Desktop/BSE/bse7/original image

Output grayscale images: C:/Users/Administrator/Desktop/BSE/bse7/gray61-25

extraction61-25

BSE › bse7 › original image

BSE › bse7 › gray61-25

BSE › bse7 › gray61-25

Fig. 6 How to finish the conversion from binary images of grains extraction into gray/color ones

- **"Manual operations" utility**

  ➢ **Open** the binary or gray/color image of grain extraction (Fig. 5d or Fig. 6e).

  "**Repair" includes**

  ➢ **Erase** can be explored to separate those connected grains by creating lines. For example, set the width of line and "Create lines" as those red lines in Fig 8a, and then click "Erase" and those connected grains are separated in Fig. 8b. **Erase** can also remove some extra of grains by creating polygen as shown in Fig. 9.

  ➢ **Gray/Color Fill** is to fill some holes inside grains according to their corresponding places in the original image. For example, the user should firstly input the original grayscale or color image by "Input original image", then click button "Gray/Color Fill", and then create polygons to enclose those blank holes inside grains (Fig. 10a), and finally click "Gray/Color Fill" so that those blank holes inside are filled according to the color or gray of the original image in their corresponding places (Fig. 10b).

  ➢ **Binary Fill** is to fill the holes inside grains just in binary, and it is not necessary to input the original gray/color images for this utility.

- **Save as** is to save the modified image as a new file once the first modification (e.g., screen, or erase, or fill) is finished.
- **Save** is to save each new modification based on the last saved file.

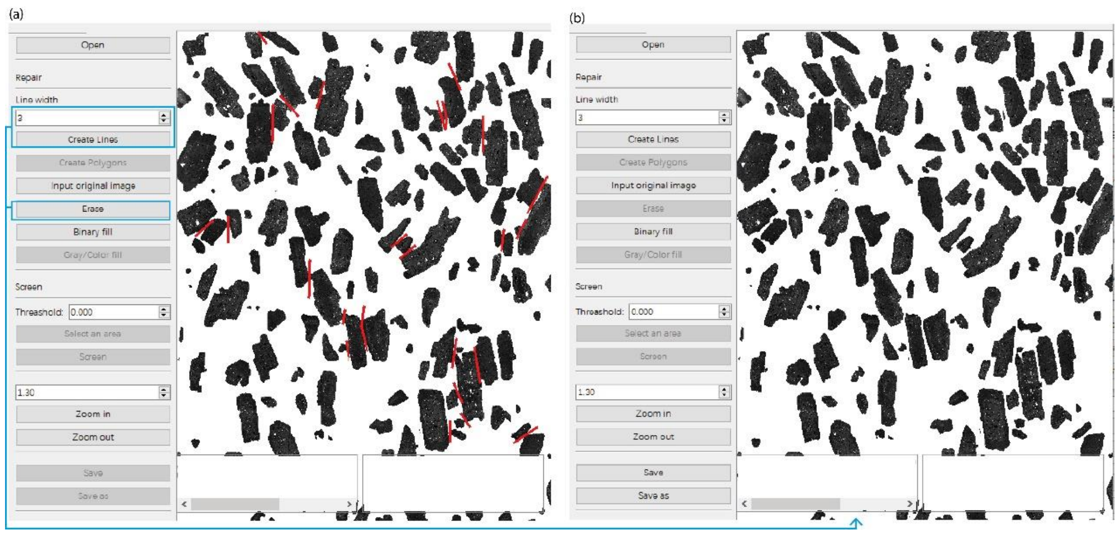


Fig. 7 How to separate connected grains by "Erase"

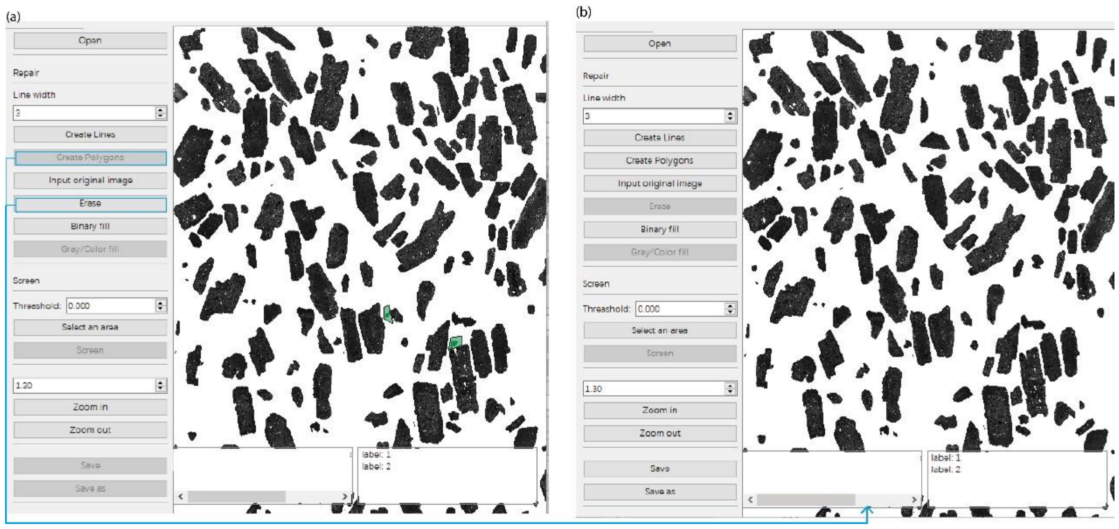
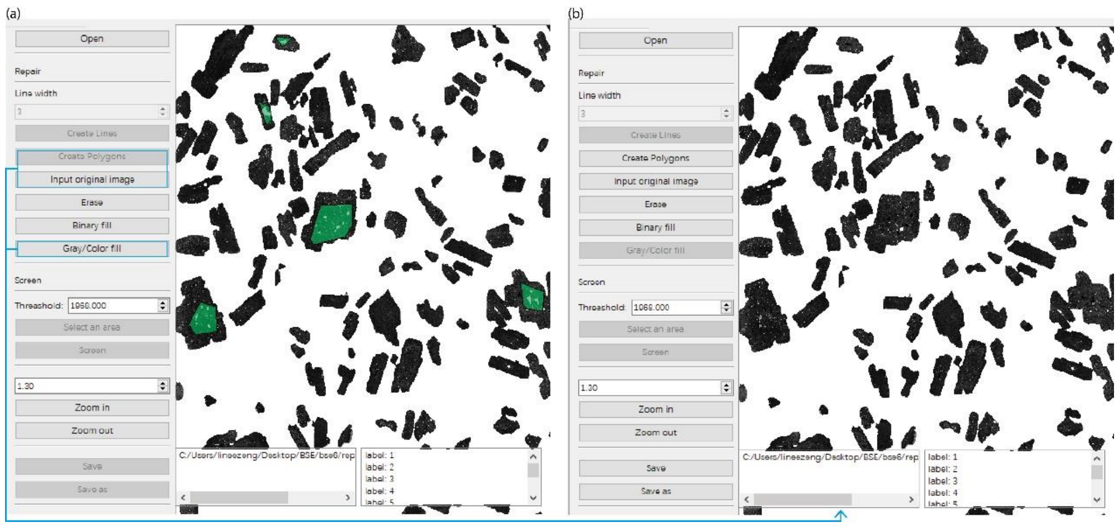


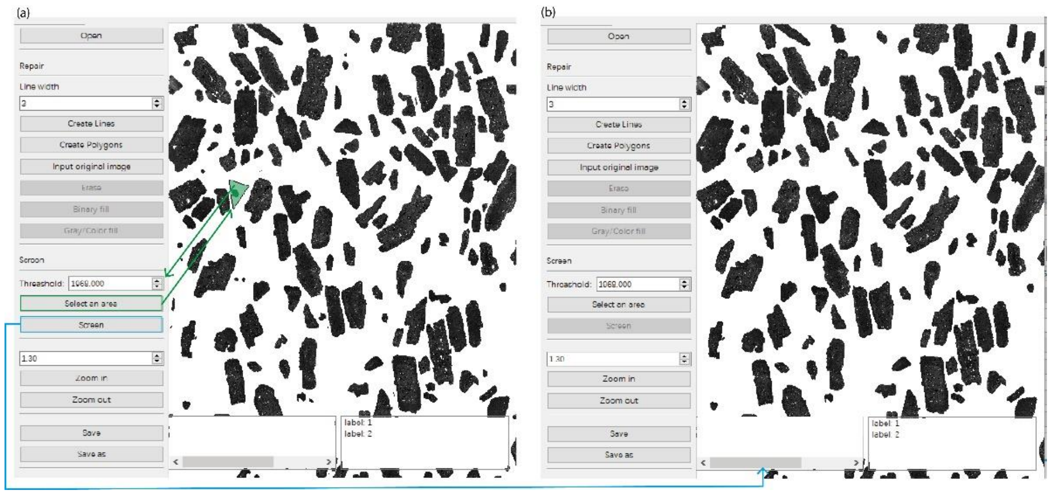
Fig. 8 How to remove some extra of grains by "Erase"



Line width: 3
Threshold: 1068.000
1.30
C:/Users/lineezeng/Desktop/BSE/bse6/rep
label: 1
label: 2
label: 3
label: 4
label: 5

Fig. 9 How to fill the holes inside grains by "Gray/Color Fill"

➢ **Screen** is to screen out those grains whose areas are below or equal to the area of the chosen grain. For example, the user can enclose a grain using "Select an area". This utility will automatically calculate the area of that enclosed grain inside the semi-transparent green triangle and show its value on the blank of "Threshold" (Fig. 10a), and then click "Screen" to screen out all grains with areas smaller than or equal to that threshold value (Fig. 10b).



Line width: 3
Threshold: 1068.000
1.30
label: 1
label: 2

Fig. 10 How to screen out

- **“Separate grains to individual images” utility**

After finishing all modifications in the step of “Manual operations”, use the resultant image (e.g., “plg.png” in Fig. 11b) as the input of this function (Fig. 11a), and create a folder (Fig. 11c) for outputting all single-grain images. After clicking “Run”, the resultant single-grain images are shown as Fig. 11d.

However, in these single-grain images, there are a few ones that may actually have two or more grains connected. Therefore, the user can simultaneously open those mistaken single-grain images in “Manual Operations”, and separate those connected grains using “Erase” one image by one image (see the below subpanel in Fig. 12a). Please “Save” (Fig. 12b) each image once finishing operation on them.

After finishing operations to all mistaken single-grain images and save the modifications on the mistaken images, the users can re-execute the function of “Separate grains to individual images”. For example, there is 622 single-grain images in the folder of “singles” after the first execution (Fig. 10d). After re-modifying those mistaken single-grain images, the users should take the folder of “singles” as the input, and create a new folder of “singles2” to save the re-execution output (Fig. 11c). Re-click “Run”, and the new output is shown in the new folder containing 634 single-grain images (Fig. 11d).

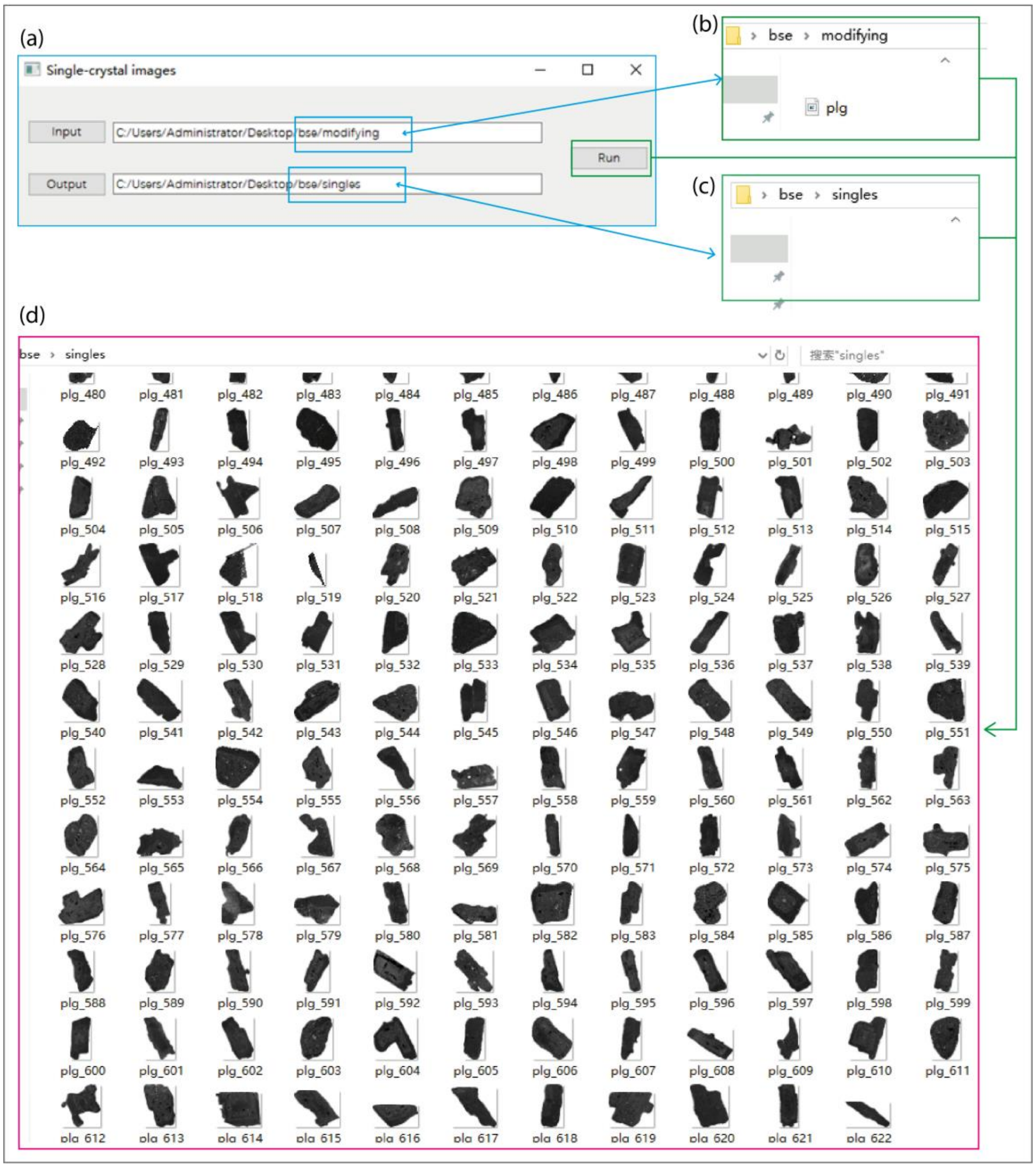
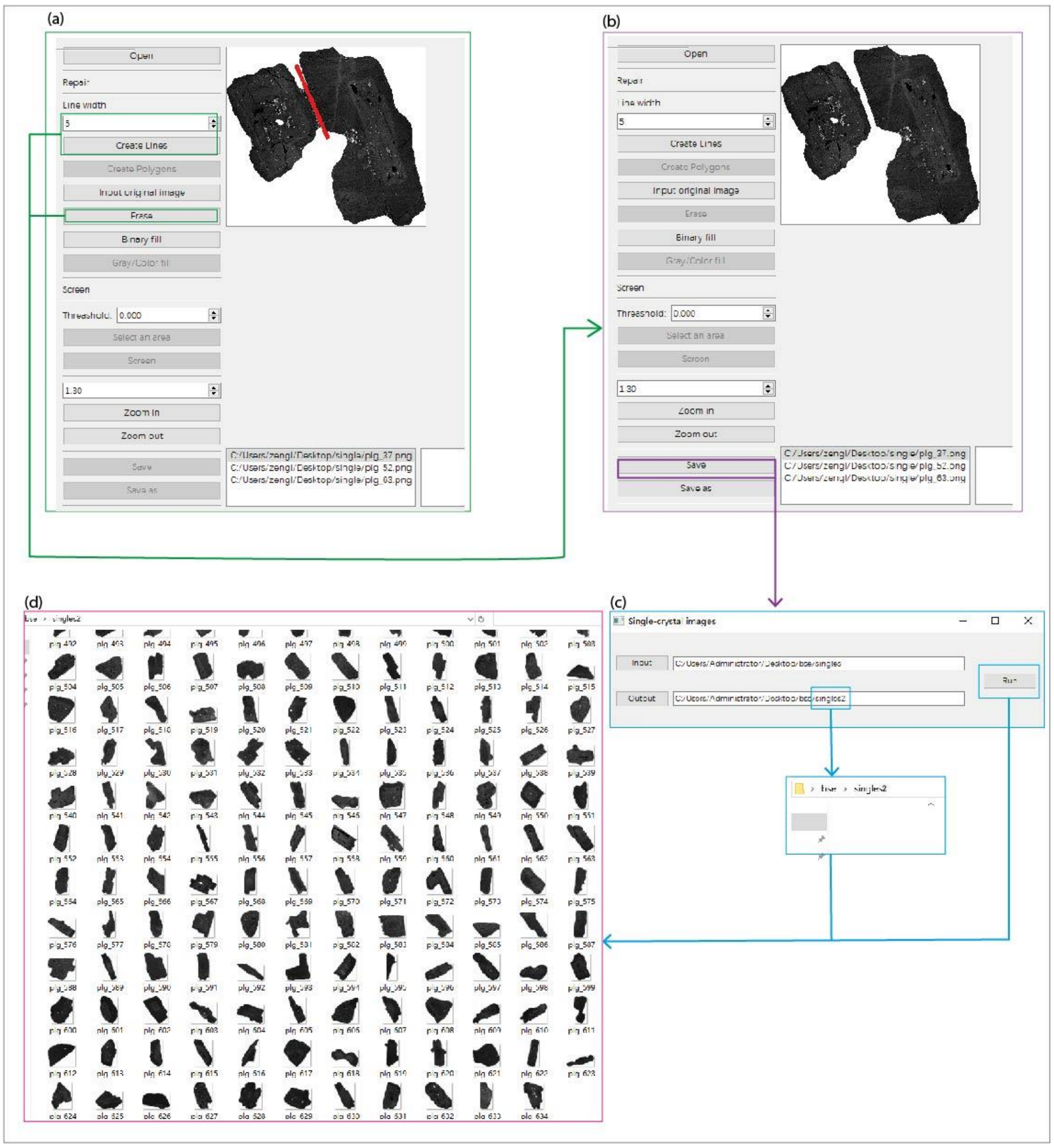

(a)
Single-crystal images
Input
C:/Users/Administrator/Desktop/bse/modifying
Output
C:/Users/Administrator/Desktop/bse/singles
Run
(b)
bse › modifying
plg
(c)
bse › singles
(d)
bse › singles
搜索"singles"
plg_480 … plg_622

Fig. 11



Line width: 5

Threshold: 0.000

1.30

C:/Users/zengl/Desktop/single/plg_37.png
C:/Users/zengl/Desktop/single/plg_52.png
C:/Users/zengl/Desktop/single/plg_63.png

Input: C:/Users/Administrator/Desktop/bse/singles

Output: C:/Users/Administrator/Desktop/bse/singles2

Fig. 12 How to re-execute “Separate grains to individual images”

- **“Scale Calibration” utility**

Open an image with scale bar, click “Draw scale” to draw one straight line along the scale bar and their lengths are equal, and then it will automatically come out the number of pixels of this scale bar. The result will be used in the next step of Grain morphologic statistics to calculate morphologic parameters of each grain.

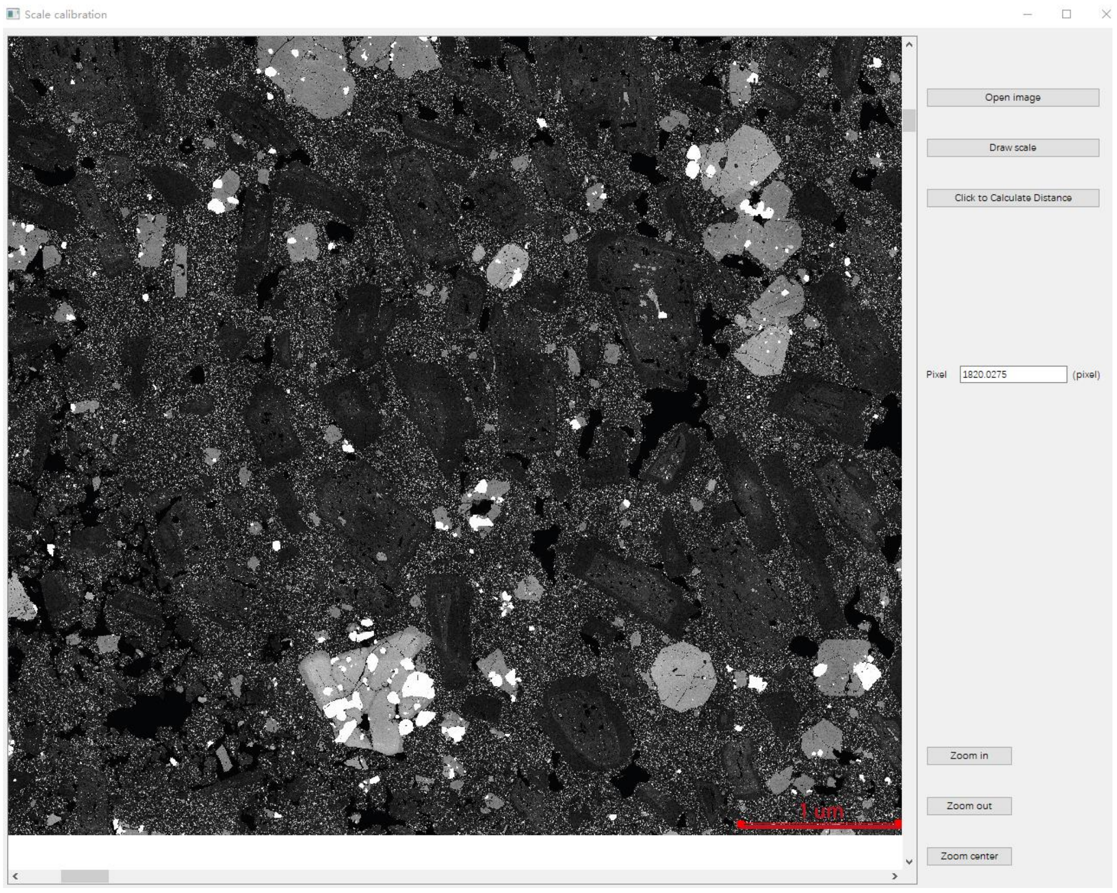
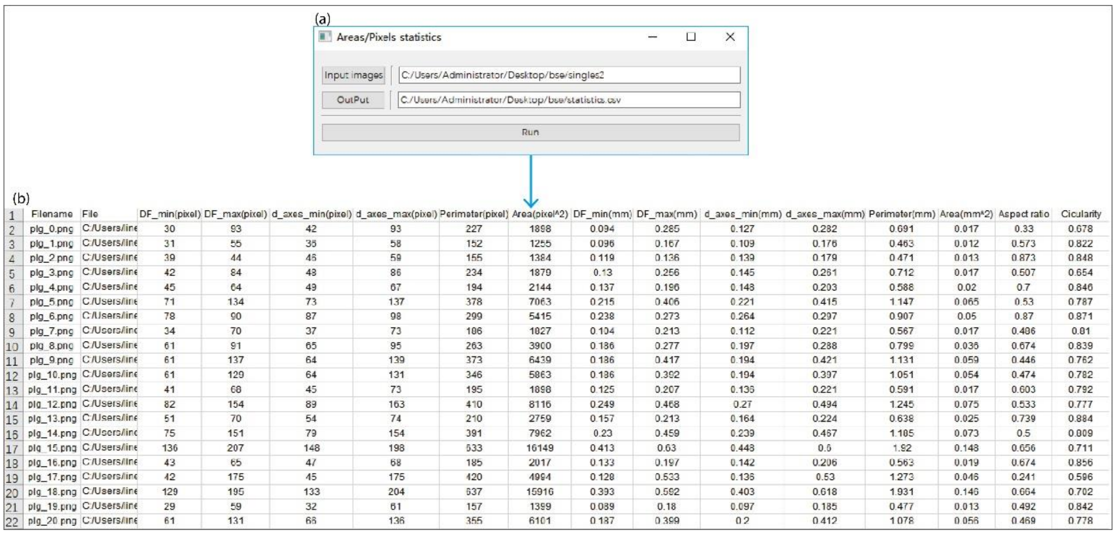


Fig. 13 Scale calibration

- **"Grain morphologic statistics" utility**

Use the folder of single-grain images as the input, and then name a "*.csv" file as output as shown in Fig. 14a. The statistical results include pixel area, pixel perimeter, pixel Feret's diameters, pixel length and width, real area, real perimeter, real Feret's diameters, real length and width, circularity, and ratio aspect for grains in single-grain images as shown in Fig. 14b.



Input images: C:/Users/Administrator/Desktop/bse/singles2

OutPut: C:/Users/Administrator/Desktop/bse/statistics.csv

1	Filename	File	DF_min(pixel)	DF_max(pixel)	d_axes_min(pixel)	d_axes_max(pixel)	Perimeter(pixel)	Area(pixel^2)	DF_min(mm)	DF_max(mm)	d_axes_min(mm)	d_axes_max(mm)	Perimeter(mm)	Area(mm^2)	Aspect ratio	Cicularity
2	plg_0.png	C:/Users/line	30	93	42	93	227	1898	0.094	0.285	0.127	0.282	0.691	0.017	0.33	0.678
3	plg_1.png	C:/Users/line	31	55	36	58	152	1255	0.096	0.167	0.109	0.176	0.463	0.012	0.573	0.822
4	plg_2.png	C:/Users/line	39	44	46	59	155	1384	0.119	0.136	0.139	0.179	0.471	0.013	0.873	0.848
5	plg_3.png	C:/Users/line	42	84	48	86	234	1879	0.13	0.256	0.145	0.261	0.712	0.017	0.507	0.654
6	plg_4.png	C:/Users/line	45	64	49	67	194	2144	0.137	0.196	0.148	0.203	0.588	0.02	0.7	0.846
7	plg_5.png	C:/Users/line	71	134	73	137	378	7063	0.215	0.406	0.221	0.415	1.147	0.065	0.53	0.787
8	plg_6.png	C:/Users/line	78	90	87	98	299	5415	0.238	0.273	0.264	0.297	0.907	0.05	0.87	0.871
9	plg_7.png	C:/Users/line	34	70	37	73	186	1827	0.104	0.213	0.112	0.221	0.567	0.017	0.486	0.81
10	plg_8.png	C:/Users/line	61	91	65	95	263	3900	0.186	0.277	0.197	0.288	0.799	0.036	0.674	0.839
11	plg_9.png	C:/Users/line	61	137	64	139	373	6439	0.186	0.417	0.194	0.421	1.131	0.059	0.446	0.762
12	plg_10.png	C:/Users/line	61	129	64	131	346	5863	0.186	0.392	0.194	0.397	1.051	0.054	0.474	0.782
13	plg_11.png	C:/Users/line	41	68	45	73	195	1898	0.125	0.207	0.136	0.221	0.591	0.017	0.603	0.792
14	plg_12.png	C:/Users/line	82	154	89	163	410	8116	0.249	0.468	0.27	0.494	1.245	0.075	0.533	0.777
15	plg_13.png	C:/Users/line	51	70	54	74	210	2759	0.157	0.213	0.164	0.224	0.638	0.025	0.739	0.884
16	plg_14.png	C:/Users/linc	75	151	79	154	391	7962	0.23	0.459	0.239	0.467	1.185	0.073	0.5	0.809
17	plg_15.png	C:/Users/line	136	207	148	198	633	16149	0.413	0.63	0.448	0.6	1.92	0.148	0.656	0.711
18	plg_16.png	C:/Users/line	43	65	47	68	185	2017	0.133	0.197	0.142	0.206	0.563	0.019	0.674	0.856
19	plg_17.png	C:/Users/line	42	175	45	175	420	4994	0.128	0.533	0.136	0.53	1.273	0.046	0.241	0.596
20	plg_18.png	C:/Users/line	129	195	133	204	637	15916	0.393	0.592	0.403	0.618	1.931	0.146	0.664	0.702
21	plg_19.png	C:/Users/line	29	59	32	61	157	1399	0.089	0.18	0.097	0.185	0.477	0.013	0.492	0.842
22	plg_20.png	C:/Users/line	61	131	66	136	355	6101	0.187	0.399	0.2	0.412	1.078	0.056	0.469	0.778

Fig. 14 Take statistics of numbers of pixels for each grain phenocryst