

Semi-Procedural Textures Using Point Process Texture Basis Functions

Guehl Pascal, Allègre Rémi, Dischler Jean-Michel, Benes Bedrich and Galin Eric

Replicability from Binaries

Our GITHUB website: <https://github.com/ASTex-ICube/semiproctex>
contains the original paper, its supplemental materials, source code and binaries.

I. Retrieving the supplemental material

DOWNLOAD the supplemental material archive:

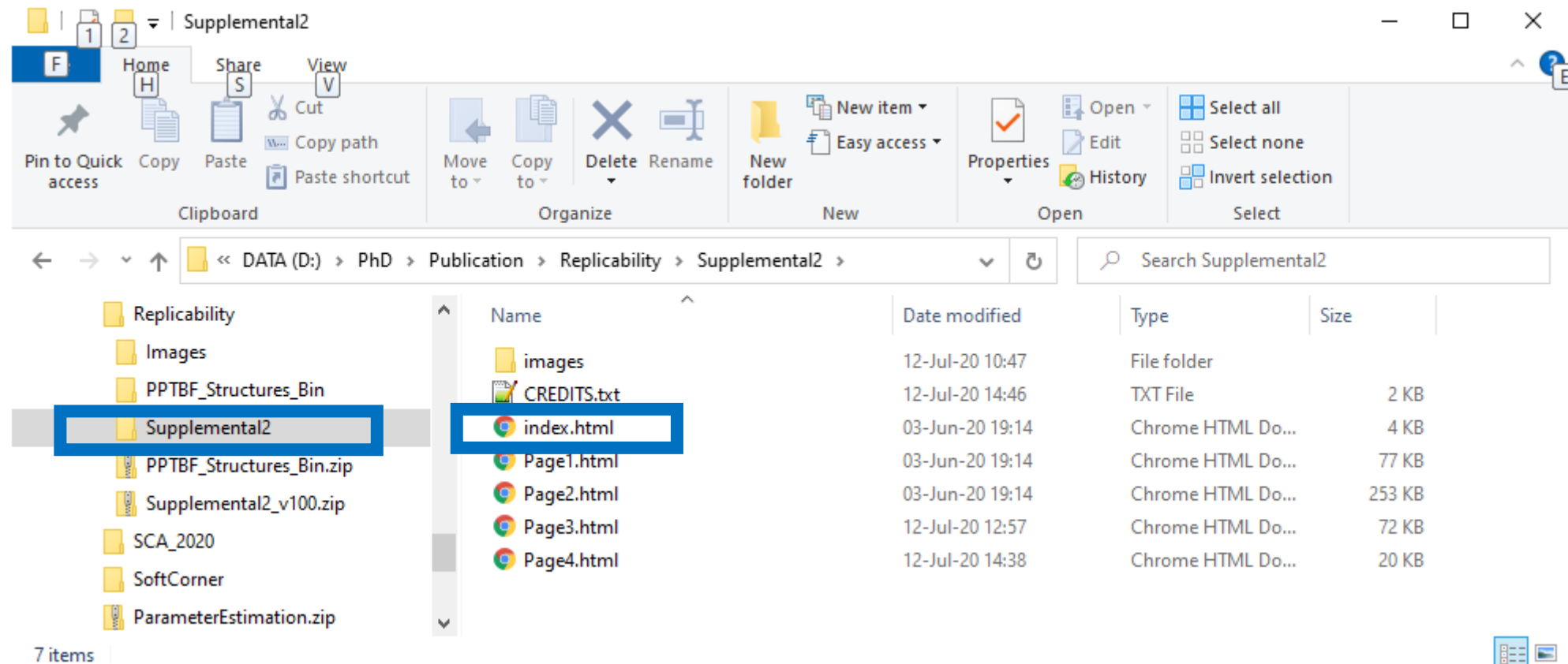
http://igg.unistra.fr/People/semiproctex/data/Supplemental2_v100.zip

I. Retrieving the supplemental material

UNZIP file: [Supplemental2_v100.zip](#)

GO TO directory: [yourPATH\Supplemental2](#)

OPEN webpage (double-click): [yourPATH\Supplemental2\index.html](#)



I. Retrieving the supplemental material

You should see the webpage on the right.

CLICK ON table of content: [1. Parameters estimation](#) to see our database of 147 parameter files (.txt format) that have been used to generate 147 binary images of our model of stochastic procedural structures.

TABLE COLUMNS:

- LEFT: input image (user provided)
- MIDDLE: image procedurally generated by the program
- RIGHT: parameter file (.txt) used to generate MIDDLE image

The figure 11 of our paper (below) is a collection of 12 of these generated images.

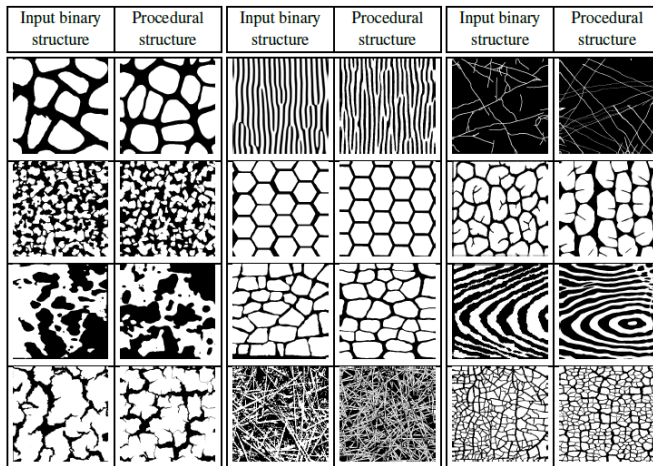
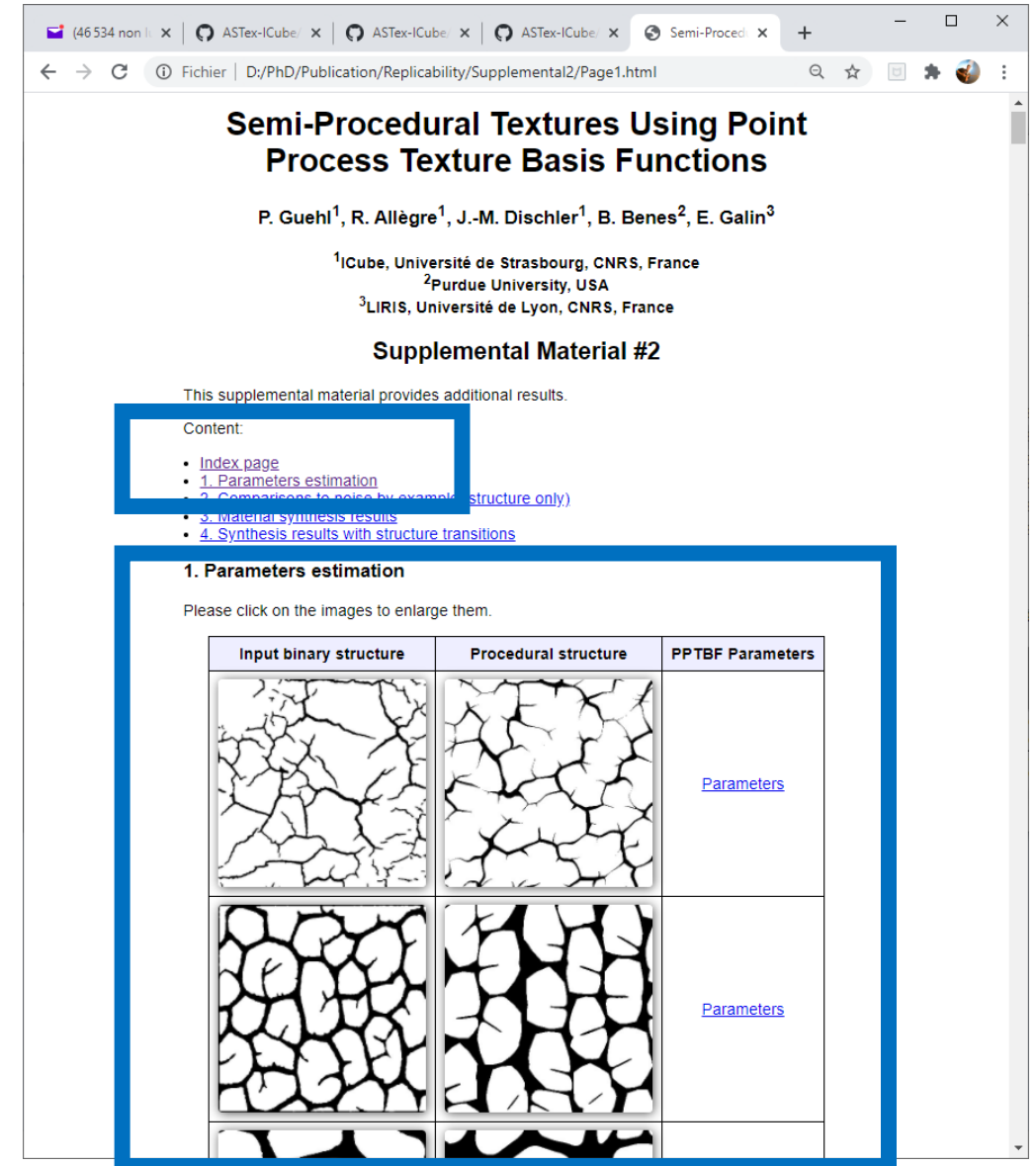


Figure 11: Evaluation of the capability of PPTBF to produce natural structures (we use segmented images on left): parameters were estimated by querying a collection of 450k samples and by applying refinement.



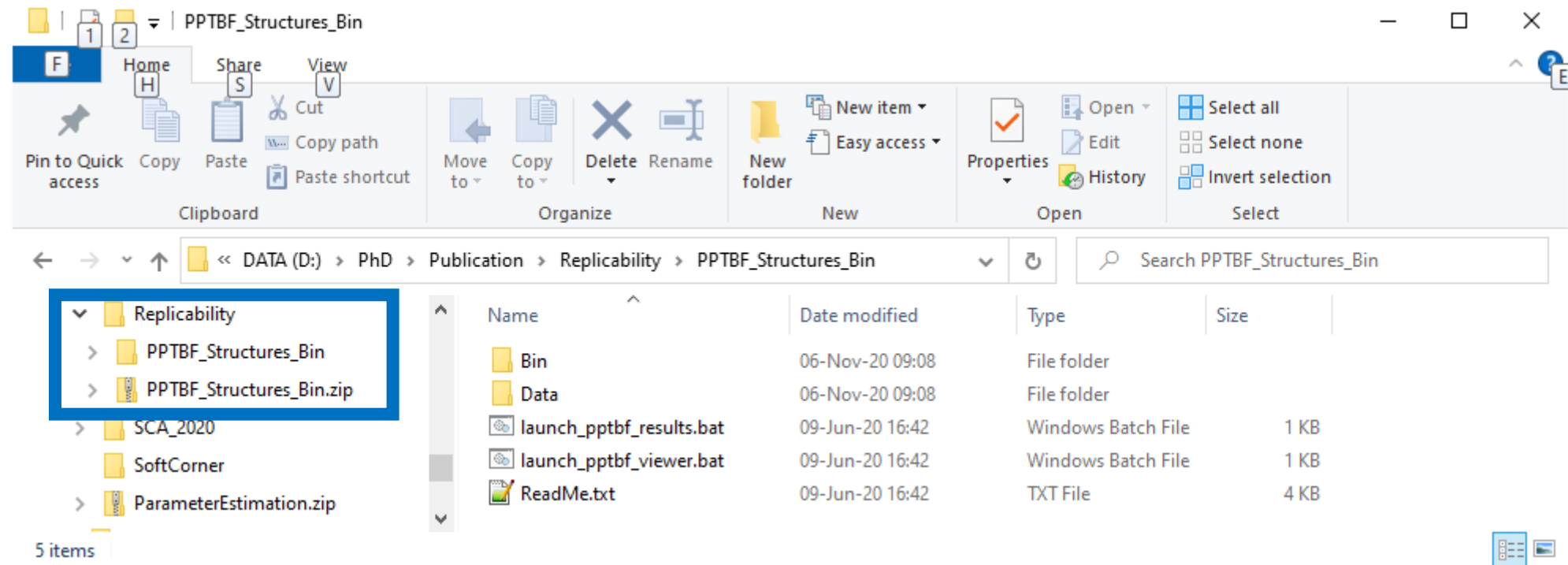
II. Retrieving the binaries

DOWNLOAD precompiled binaries archive: [PPTBF_Structures_Bin.zip](http://igg.unistra.fr/people/semiproctex/PPTBF_Structures_Bin.zip)
http://igg.unistra.fr/people/semiproctex/PPTBF_Structures_Bin.zip

II. Retrieving the binaries

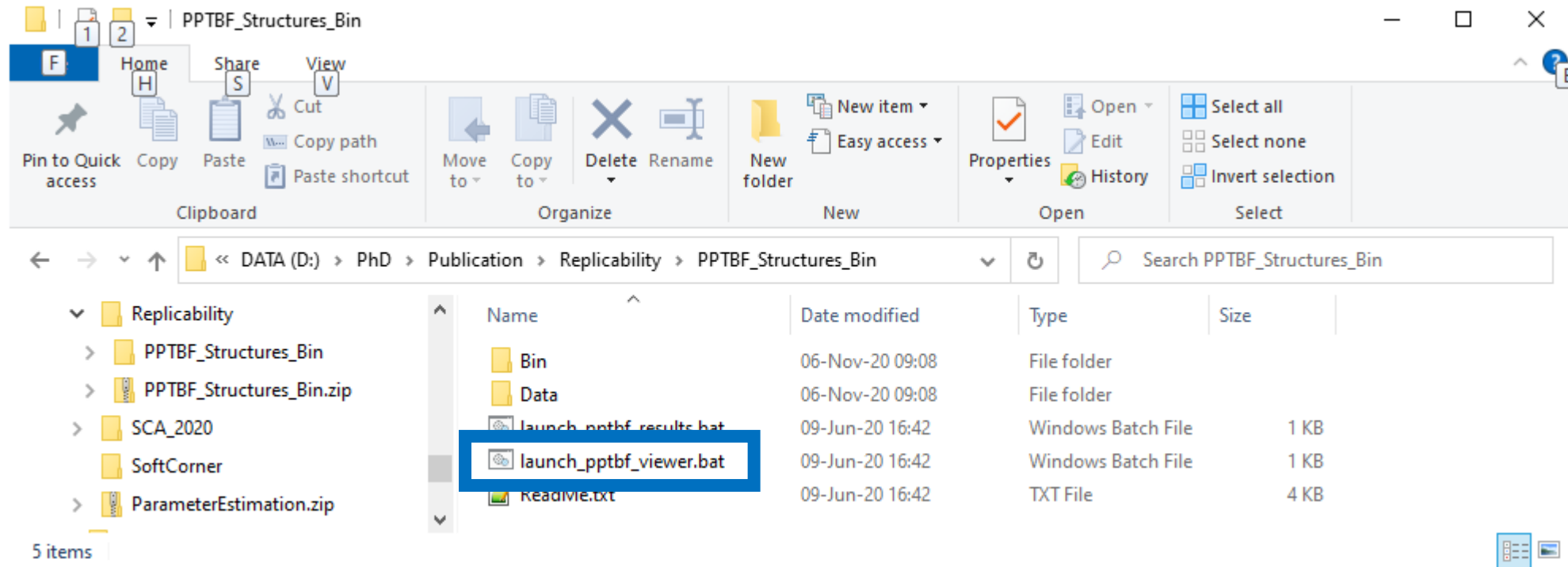
UNZIP file: [PPTBF_Structures_Bin.zip](#)

GO TO directory: [yourPATH\PPTBF_Structures_Bin](#)



III. Check Graphics Program

LAUNCH the 3D graphics viewer with SCRIPT: [launch_pptbf_viewer.bat](#)



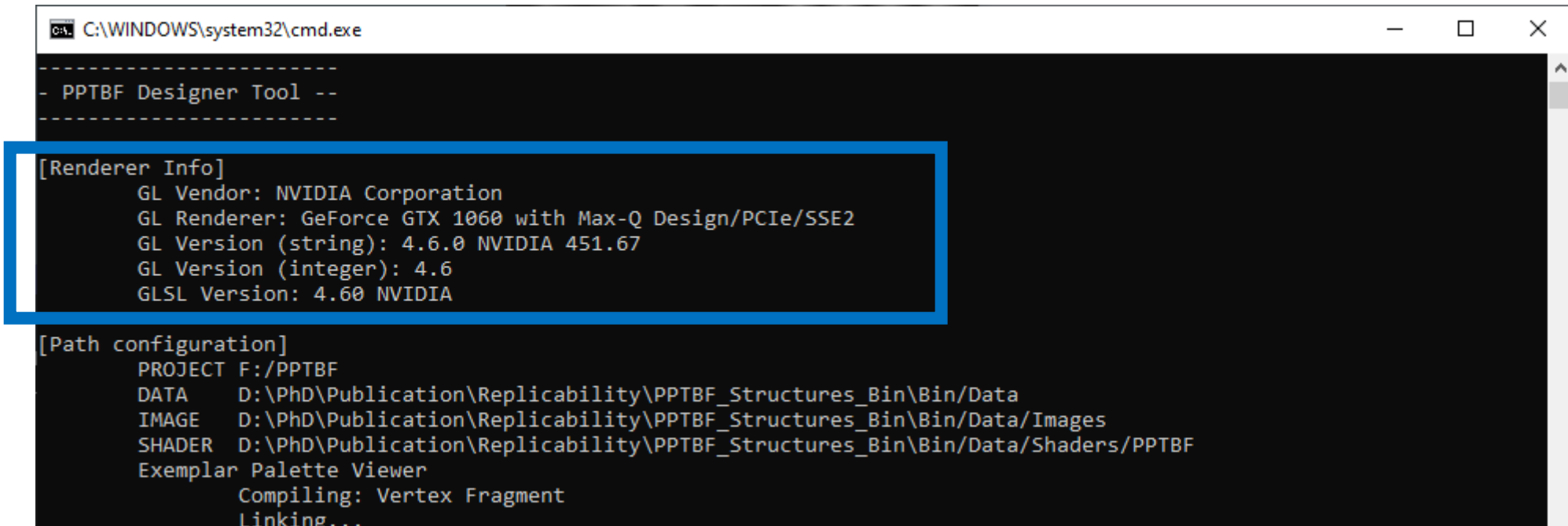
III. Check Graphics Program

a DOS window opens and LOG some info

Check “Renderer Info”: you MUST see your graphics card (GPU), but NOT an “integrated” one such as INTEL

NOTE: Program requires a GPU close to NVIDIA GTX 1060

IF “integrated”, the viewer will not launch or crash, see next slide



```
C:\WINDOWS\system32\cmd.exe

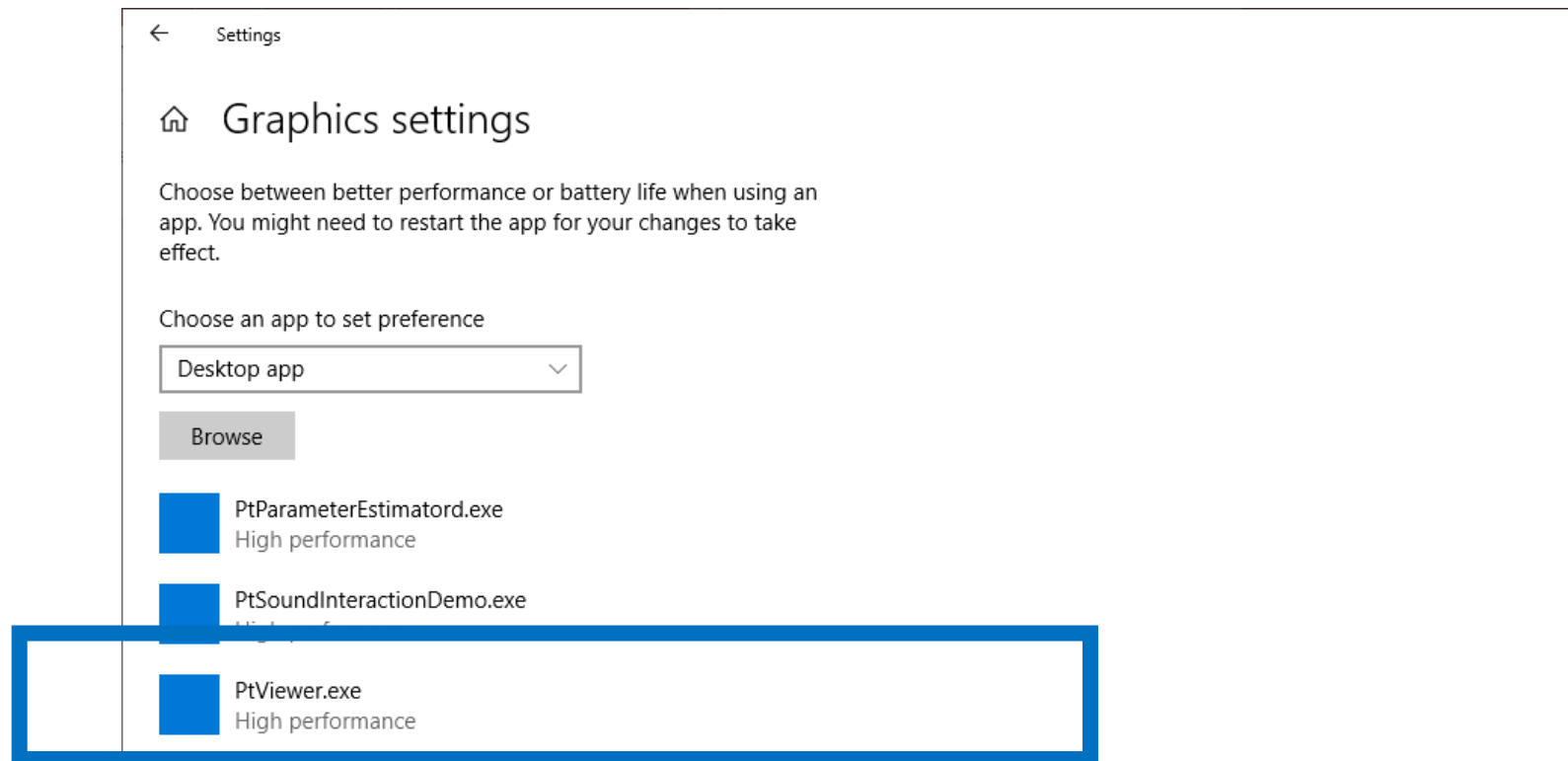
-----
- PPTBF Designer Tool --
-----

[Renderer Info]
  GL Vendor: NVIDIA Corporation
  GL Renderer: GeForce GTX 1060 with Max-Q Design/PCIe/SSE2
  GL Version (string): 4.6.0 NVIDIA 451.67
  GL Version (integer): 4.6
  GLSL Version: 4.60 NVIDIA

[Path configuration]
  PROJECT F:/PPTBF
  DATA   D:\PhD\Publication\Replicability\PPTBF_Structures_Bin\Bin\Data
  IMAGE   D:\PhD\Publication\Replicability\PPTBF_Structures_Bin\Bin\Data/Images
  SHADER  D:\PhD\Publication\Replicability\PPTBF_Structures_Bin\Bin\Data/Shaders/PPTBF
  Exemplar Palette Viewer
          Compiling: Vertex Fragment
          Linking...
```


III. Check Graphics Program

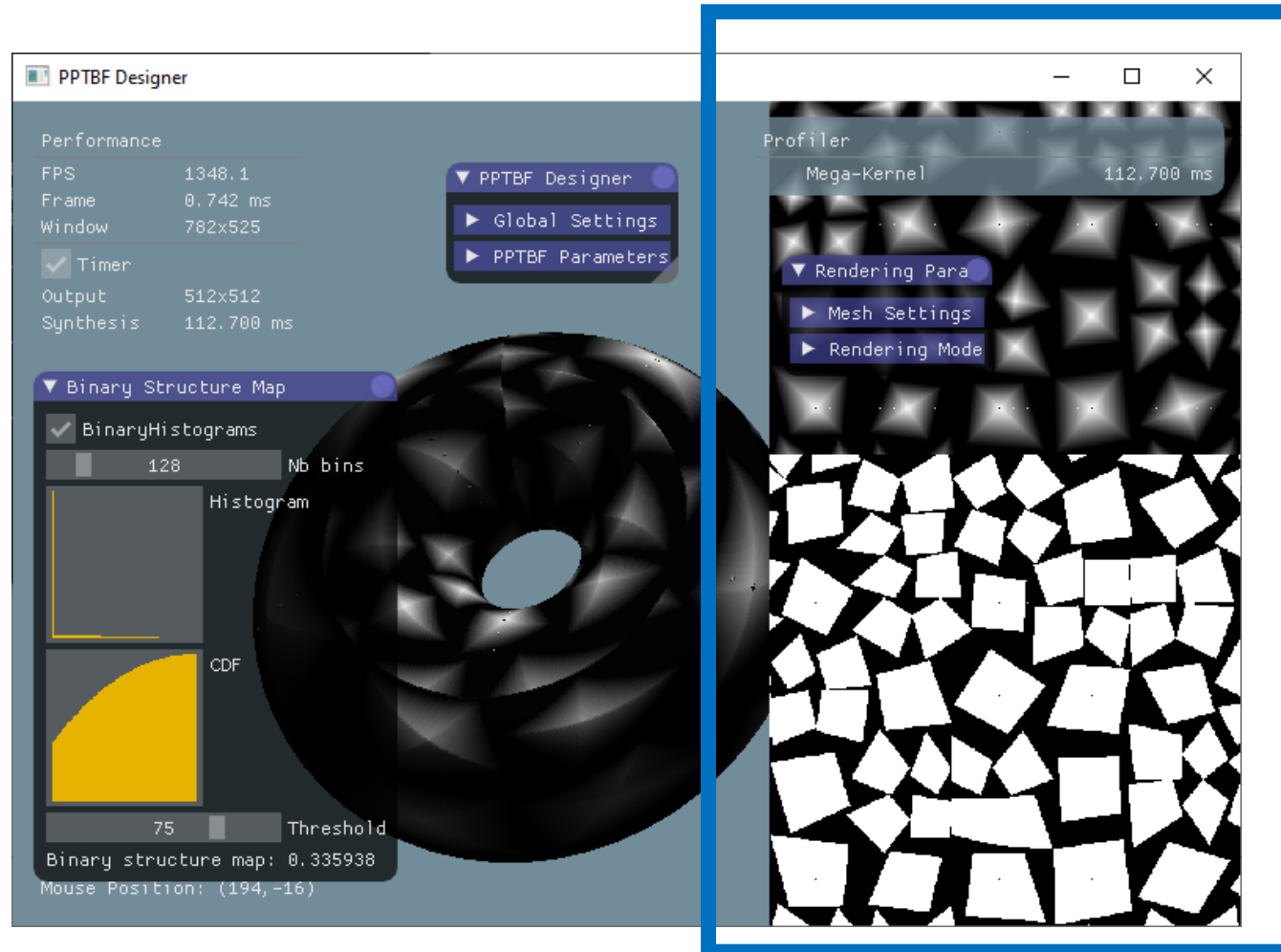
IF using “integrated” device by default (this is classical on LAPTOP to optimize battery), the two programs `yourPath\PPTBF_Structures_Bin\Bin\PtViewer.exe` and `yourPath\PPTBF_Structures_Bin\Bin\PtBDDGenerator.exe` have to be added in the list of programs in the “Graphics settings” preferences of Windows 10 to launch it with High Performance



III. Check Graphics Program

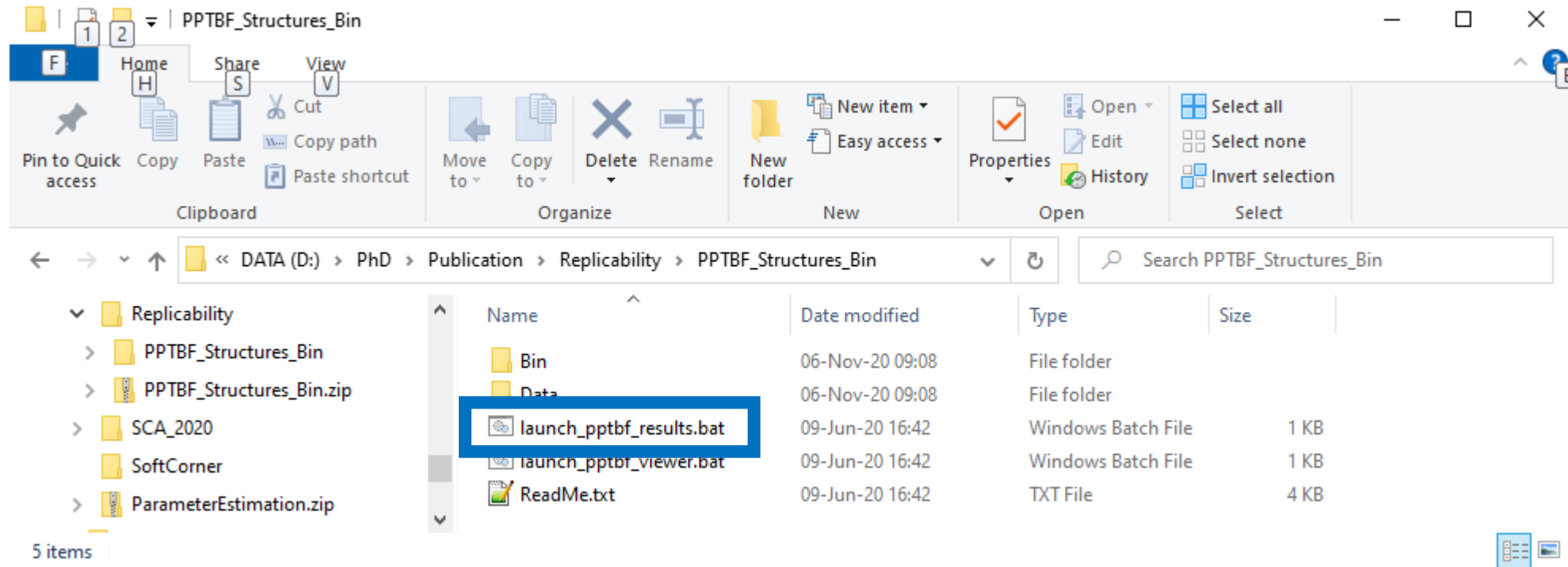
If program works, you should see a 3D graphics viewer with 2D images on the right

- these images are our generated “stochastic procedural structures”
- this is the kind of images we are going to generate in batch mode (with script) to reproduce all our related results



IV. Launch Results generation

LAUNCH the SCRIPT to generate all (supplemental) results : [launch_pptbf_results](#)

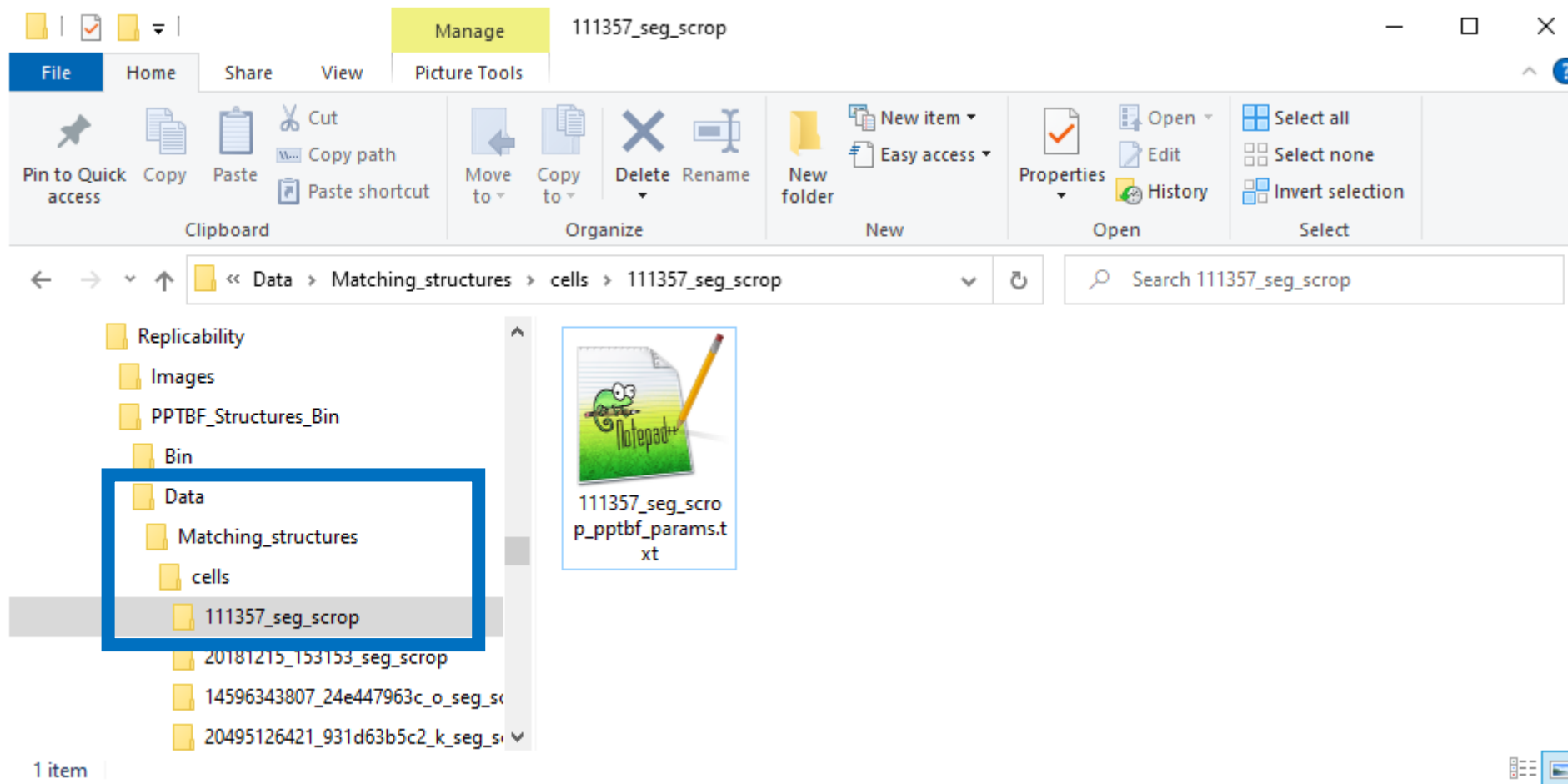


IV. Launch Results generation

The program will generate 2 images for each parameter file located in subfolders of the data directory:

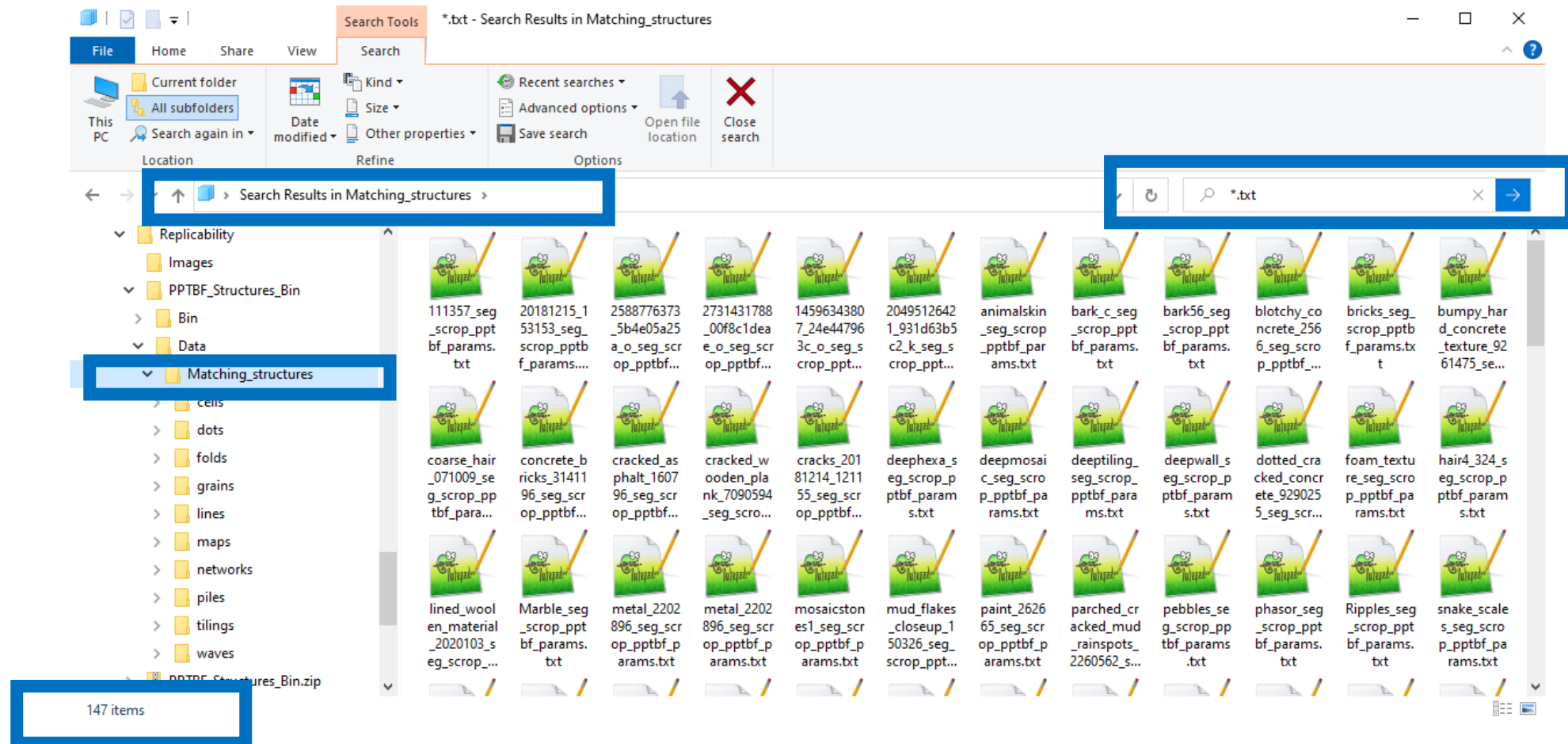
`yourPath\Data\Data\Matching_structures`

Data is classified by type (cells, lines, dots, etc...) in subdirectories, with 1 directory per parameter file (as below)



IV. Launch Results generation

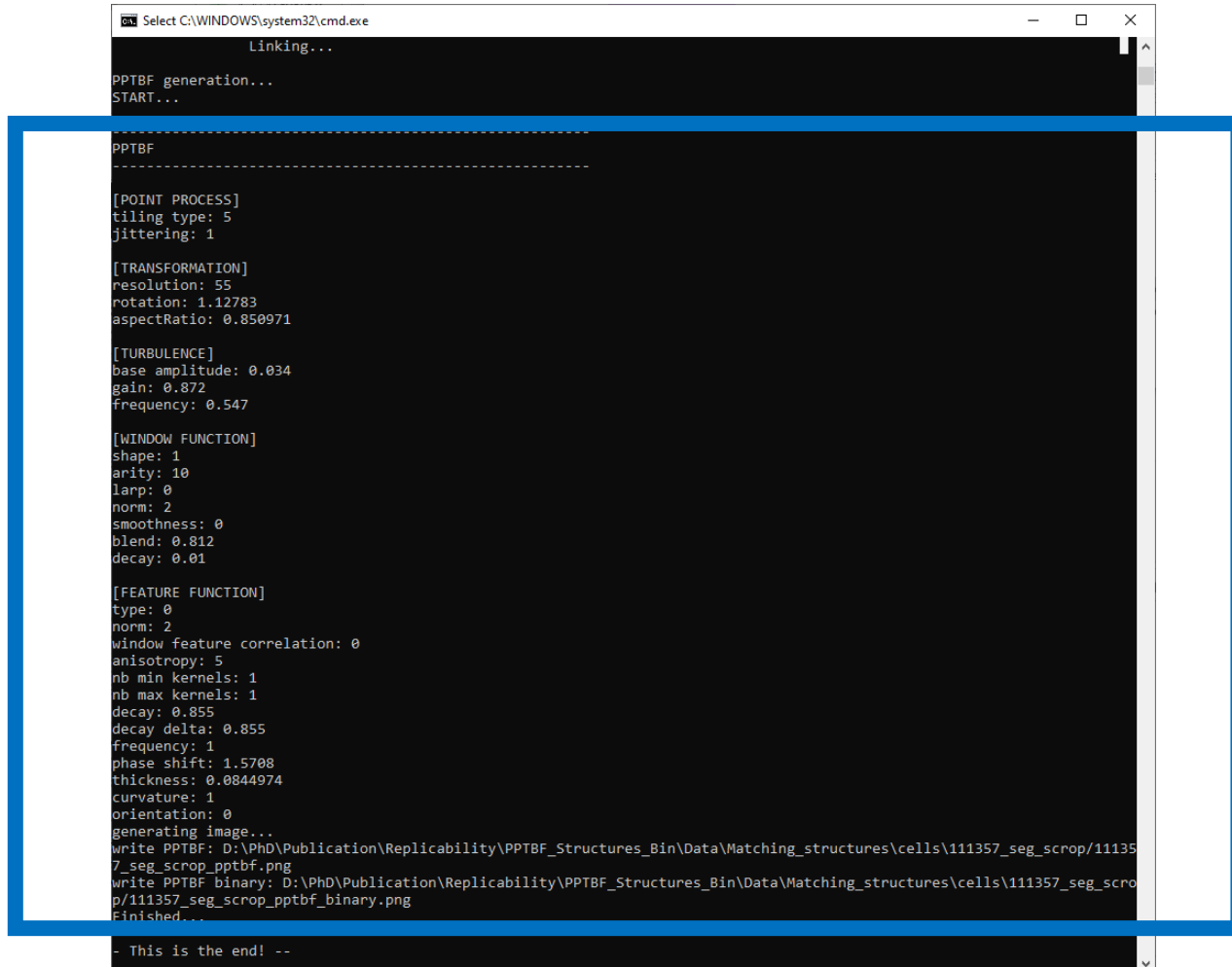
ALL DATA is classified by type (cells, lines, dots, etc...) in subdirectories
TO SEE ALL, just search for parameter files *.txt
Our supplemental material contains 147 files (see below):



IV. Launch Results generation

a DOS window opens and LOG some info

For each parameter file, the program reads it and displays its parameters info, then generate 2 images (.png format) and write them in the same directory

A screenshot of a DOS command window titled "Select C:\WINDOWS\system32\cmd.exe". The window displays the output of a program. The text is as follows:

```
Linking...
PPTBF generation...
START...

PPTBF
-----

[POINT PROCESS]
tiling type: 5
jittering: 1

[TRANSFORMATION]
resolution: 55
rotation: 1.12783
aspectRatio: 0.850971

[TURBULENCE]
base amplitude: 0.034
gain: 0.872
frequency: 0.547

[WINDOW FUNCTION]
shape: 1
arity: 10
larp: 0
norm: 2
smoothness: 0
blend: 0.812
decay: 0.01

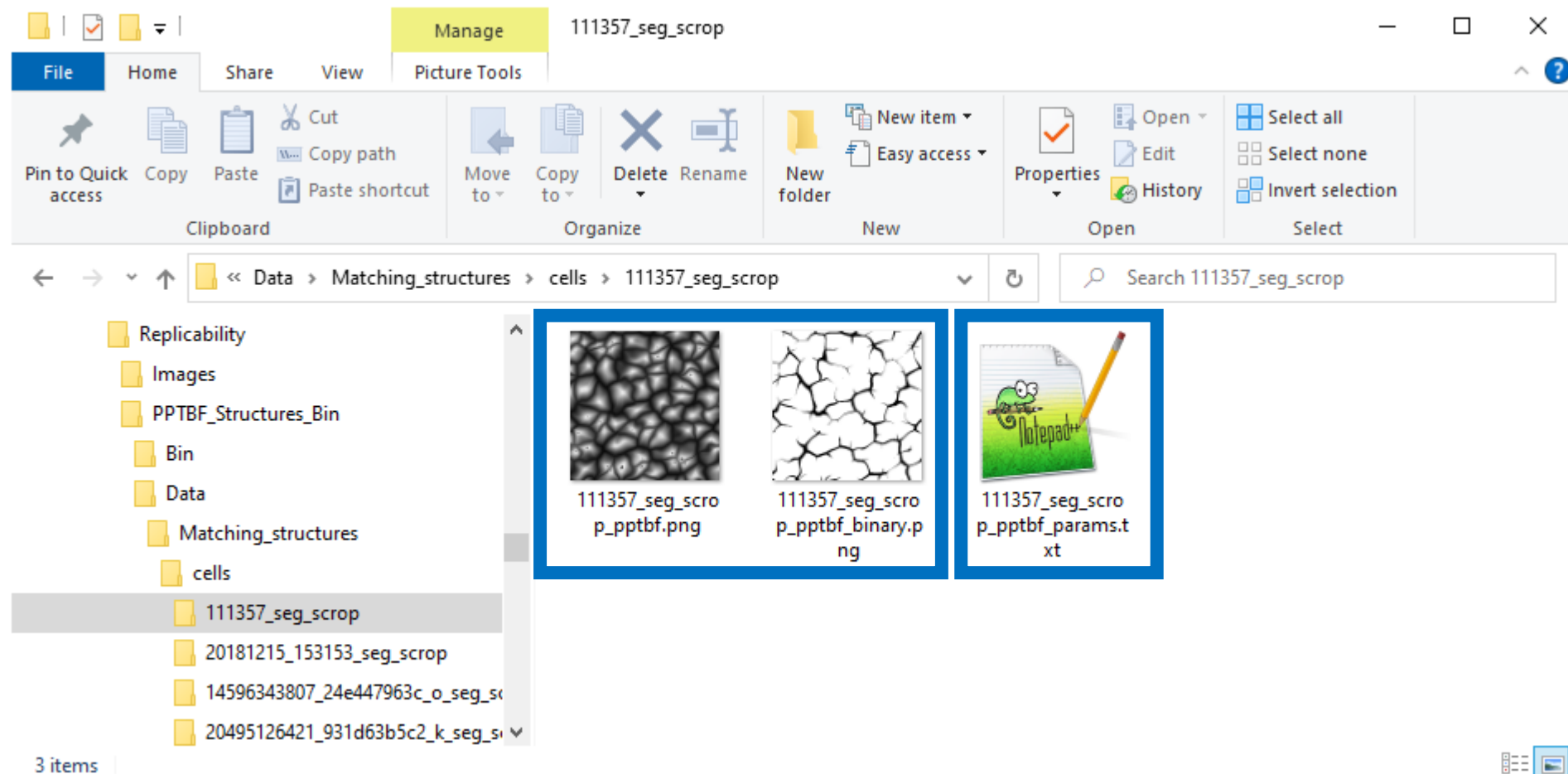
[FEATURE FUNCTION]
type: 0
norm: 2
window feature correlation: 0
anisotropy: 5
nb min kernels: 1
nb max kernels: 1
decay: 0.855
decay delta: 0.855
frequency: 1
phase shift: 1.5708
thickness: 0.0844974
curvature: 1
orientation: 0
generating image...
write PPTBF: D:\PhD\Publication\Replicability\PPTBF_Structures_Bin\Data\Matching_structures\cells\111357_seg_scrop\111357_seg_scrop_pptbf.png
write PPTBF binary: D:\PhD\Publication\Replicability\PPTBF_Structures_Bin\Data\Matching_structures\cells\111357_seg_scrop\111357_seg_scrop_pptbf_binary.png
Finished...

- This is the end! --
```

IV. Launch Results generation

If program works, you should see 2 images (.png) in the same directory for each parameter file :

- one image `xxx_pptbf.png` that is the stochastic procedural structure (grayscale)
- one image `xxx_pptbf_binary.png` that is its binary version (the one we want to compare)

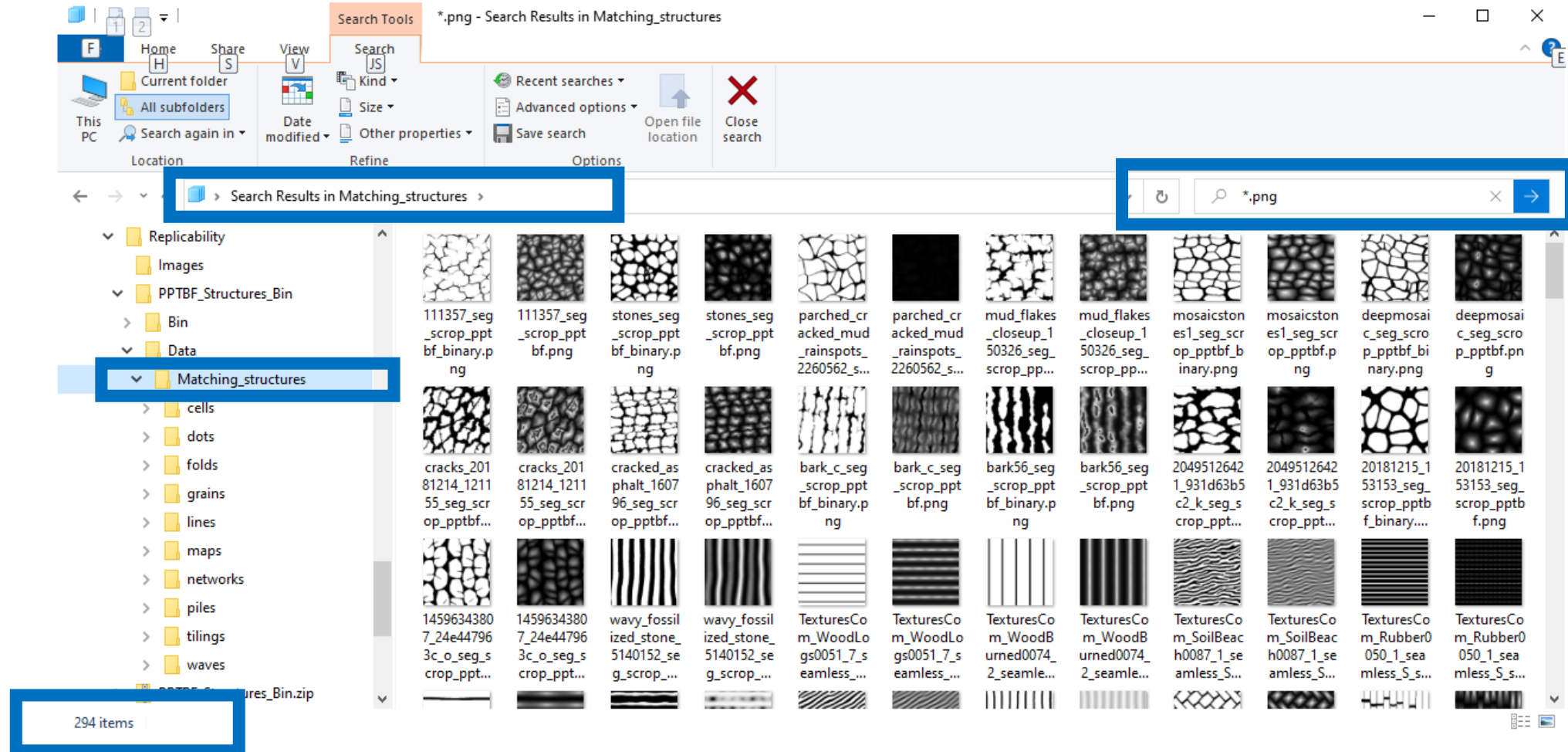


IV. Launch Results generation

ALL generated DATA is classified by type (cells, lines, dots, etc...) in subdirectories

TO SEE ALL, just search for generated images files `*.png`

Our supplemental material contains 2 images for each 147 files, so 294 images (see below):



V. Compare results

In the figure 11, you can find the following replicated images:

LEFT COLUMN (top to bottom):

- 20181215_153153_seg_scorp_pptbf_binary.png
- bumpy_hard_concrete_texture_9261475_seg_scorp_pptbf_binary.png
[NOTE: for this image, you may find a difference due to a translation in the original article]
- foam_texture_seg_scorp_pptbf_binary
- mud_flakes_closeup_150326_seg_scorp_pptbf_binary.png

MIDDLE COLUMN (top to bottom):

- phasor_seg_scorp_pptbf_binary.png
- deephexa_seg_scorp_pptbf_binary.png
- mosaicstones1_seg_scorp_pptbf_binary.png
- straw_seg_scorp_pptbf_binary.png

RIGHT COLUMN (top to bottom):

- Marble_seg_scorp_pptbf_binary.png
- 14596343807_24e447963c_o_seg_scorp_pptbf_binary.png
- whiteash_seg_scorp_pptbf_binary.png
- TexturesCom_Crackles0011_S_seg_scorp_pptbf_binary.png

[Check next slide for comparisons](#)

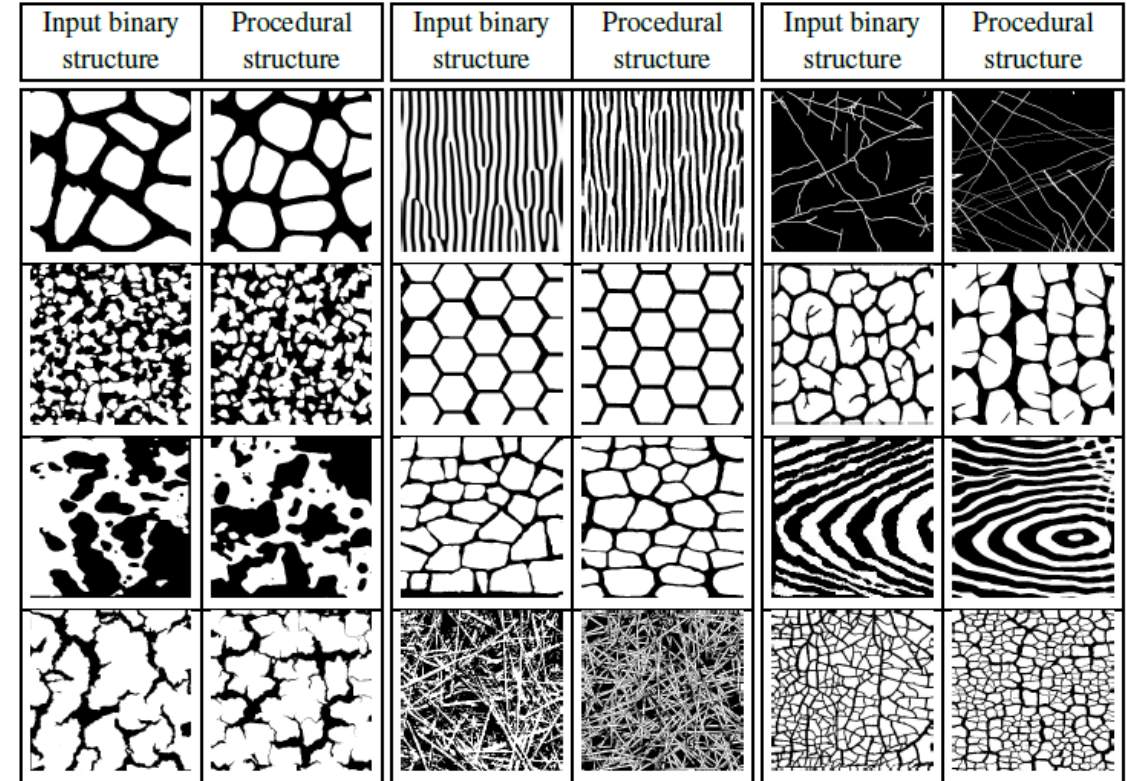


Figure 11: *Evaluation of the capability of PPTBF to produce natural structures (we use segmented images on left): parameters were estimated by querying a collection of 450k samples and by applying refinement.*

V. Compare results

Supplemental Material (extracted snapshots)

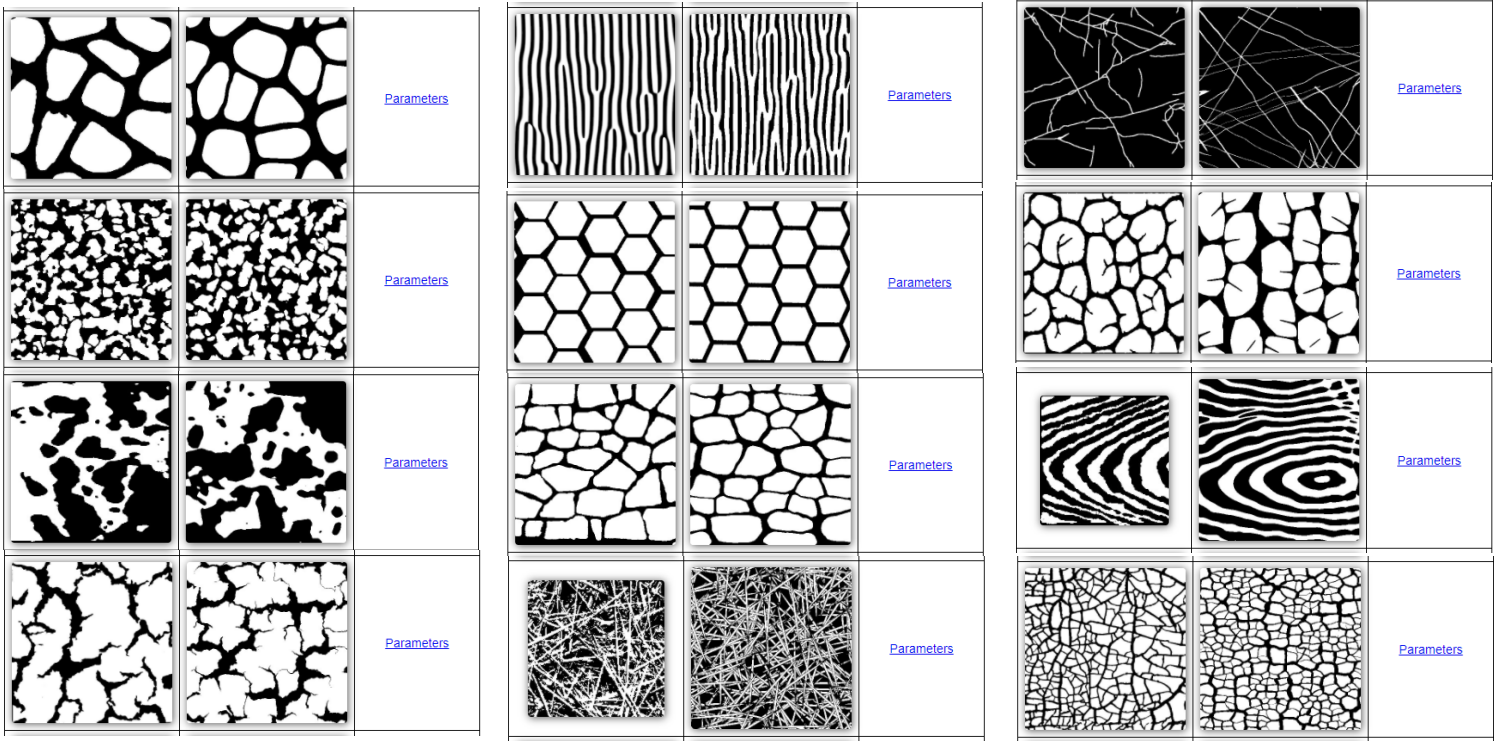


Figure 11 from article

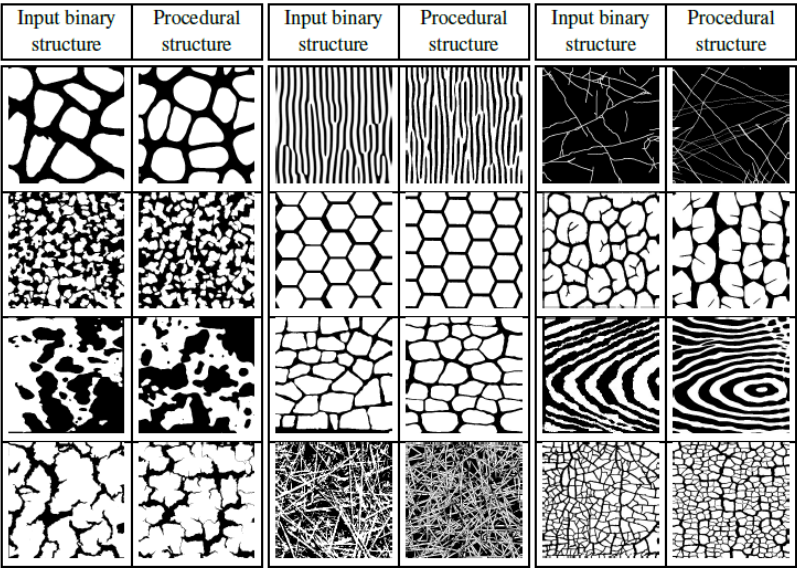


Figure 11: Evaluation of the capability of PPTBF to produce natural structures (we use segmented images on left): parameters were estimated by querying a collection of 450k samples and by applying refinement.

Generated Images: located in directory `yourPath\Data\Matching_structures\`

<code>cells\20181215_153153_seg_scrop</code>	<code>lines\phasor_seg_scrop</code>	<code>networks\Marble_seg_scrop</code>
<code>dots\bumpy_hard_concrete_texture_9261475_seg_scrop</code>	<code>tilings\deephexa_seg_scrop</code>	<code>cells\14596343807_24e447963c_o_seg_scrop</code>
<code>maps\foam_texture_seg_scrop</code>	<code>cells\mosaicstones1_seg_scrop</code>	<code>grains\whiteash_seg_scrop</code>
<code>cells\mud_flakes_closeup_150326_seg_scrop</code>	<code>piles\straw_seg_scrop</code>	<code>cells\TexturesCom_Crackles0011_S_seg_scrop</code>

IMPORTANT

On the left table, we display the directories where the 12 images to compare against figure 11 have been procedurally generated.