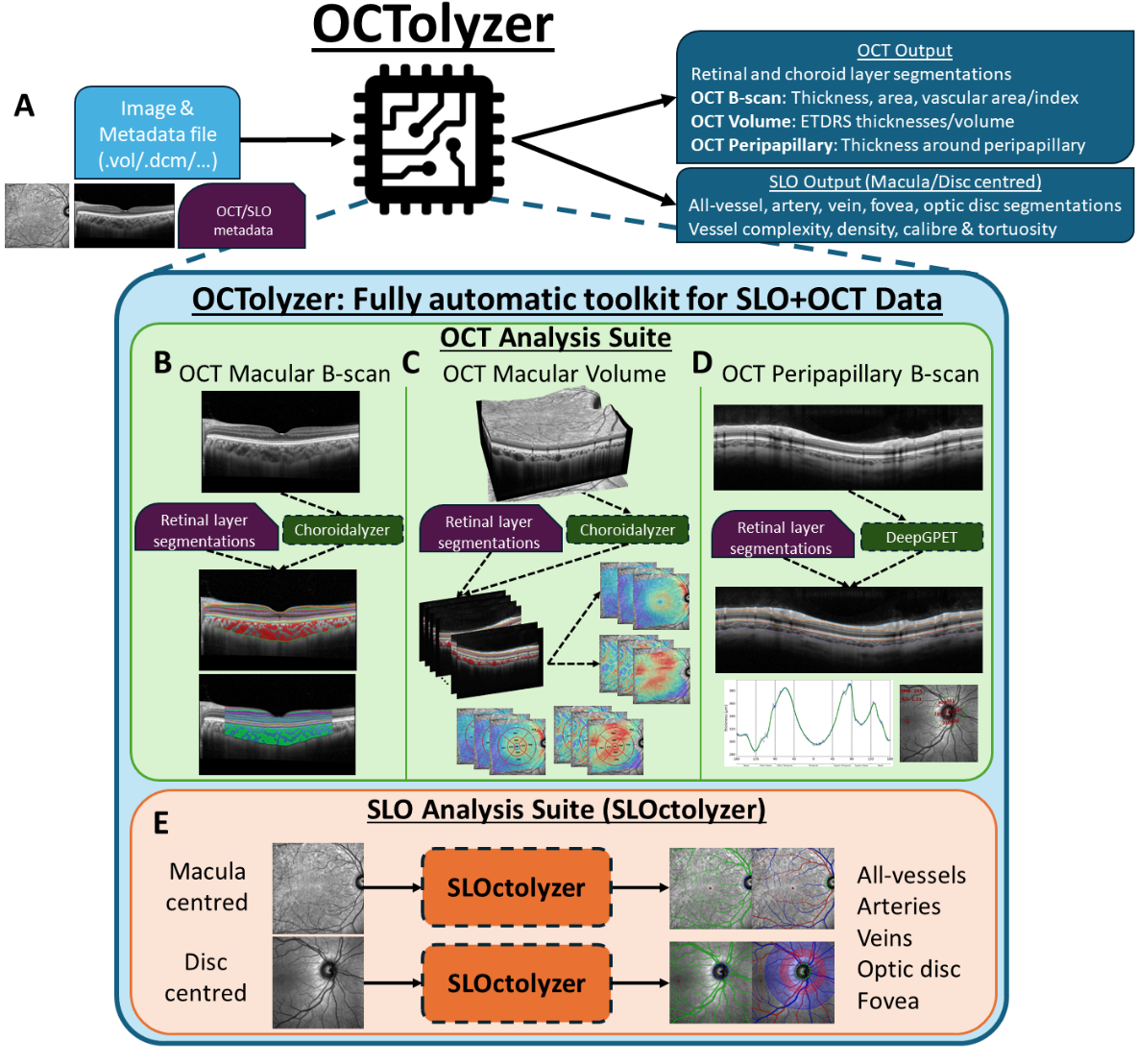
**Instructions for OCTolyzer: A fully automatic toolkit for segmentation and feature extracting in optical coherence tomography (OCT) and scanning laser ophthalmoscopy (SLO) data**

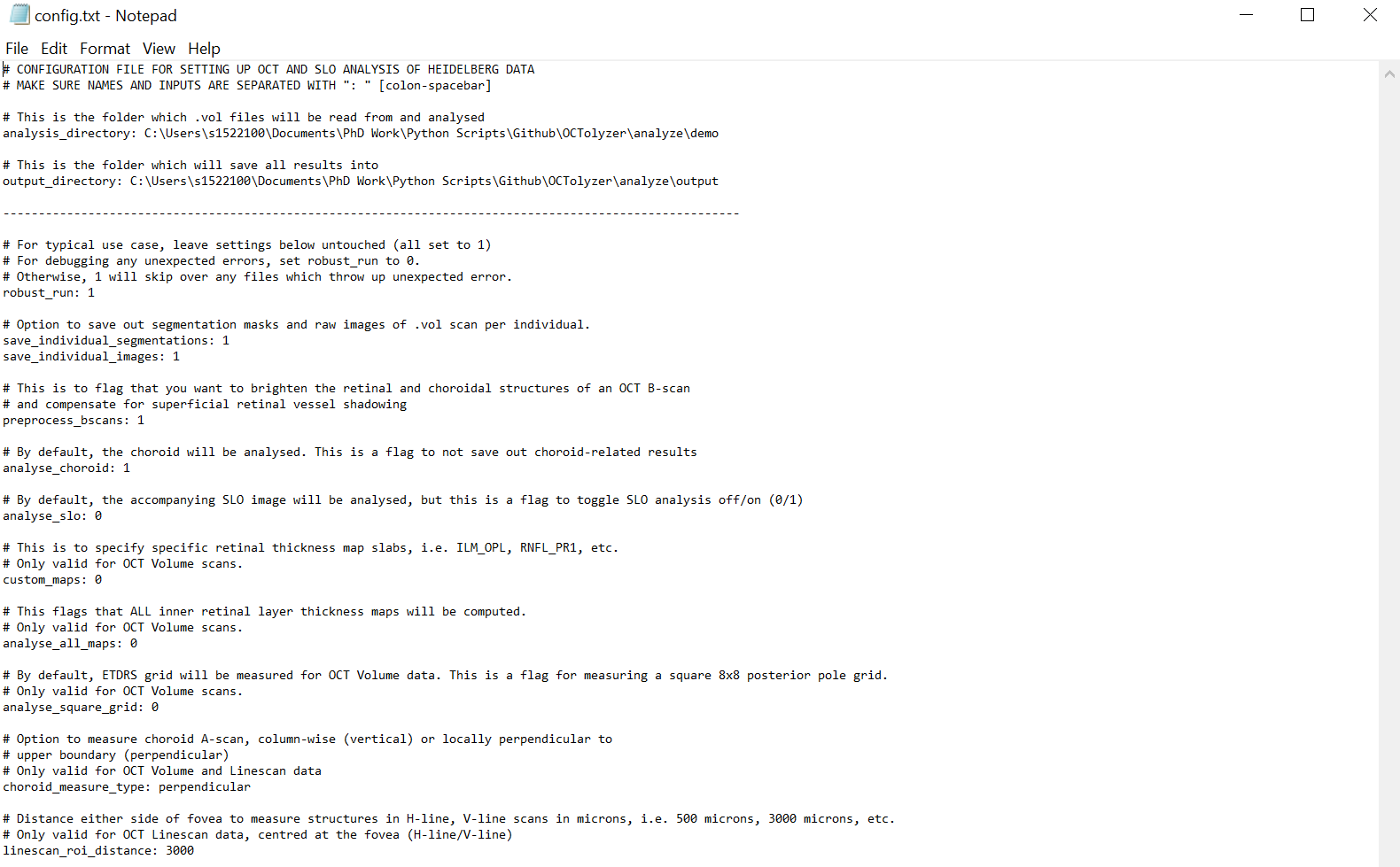
OCTolyzer is a fully automatic pipeline which is capable of fully characterising the cross sectional retinal and choroid layers in OCT images and en face retinal vessels in SLO images. The pipeline utilises fully automatic deep learning methods for segmenting these landmarks, including characterisation of the choroidal vasculature and classification of the en face retinal vessels into arteries and veins.

OCTolyzer is also capable of extracting clinically-relevant features of interest of the segmented retina and choroid. The code used to measure regional and spatial measurements from OCT images was developed in-house, while the code used to measure features of en face retinal vessels were heavily based on the code produced by [Automorph](https://tvst.arvojournals.org/article.aspx?articleid=2783477), whose codebase can be found [here](https://github.com/rmaphoh/AutoMorph).

****At present, OCTolyzer supports Heidelberg specific .VOL files which contain image and metadata.

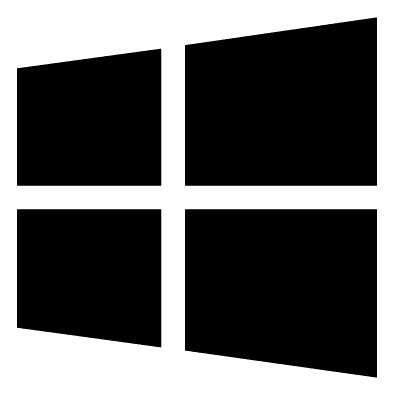
**1) GETTING STARTED FROM THE TERMINAL**

The configuration file config.txt is used in conjunction with running the software from the terminal (see below) and contains user-specified parameters. See the table below for each parameter and their definition.



|  |  |  |  |
| --- | --- | --- | --- |
| Parameter | Description | Expected value | Default value |
| analysis\_directory | Where OCTolyzer will look for .vol files. | Any valid directory path | Path\to\OCTolyzer\analyze\demo |
| output\_directory | Where the metadata, segmentations and measurements are saved per .vol file. | Any valid directory path | Path\to\OCTolyzer\analyze\output |
| robust\_run | This will ensure the software will run without fail, skipping over any files which failed to be analysed unexpectedly. Set this to 0 to debug any problems. | 0 (No) or 1 (Yes) | 1 |
| save\_individual\_segmentations | Flag to save out segmentations, a log file and images showing segmentations on the B-scan(s) and/or thickness maps superimposed on the SLO.  Might not be preferred if processing large batches | 0 (No) or 1 (Yes) | 1 |
| save\_individual\_images | This saves out the SLO and B-scan(s). | 0 (No) or 1 (Yes) | 1 |
| preprocess\_bscans | Flag to brighten B-scans and compensate for superficial retinal vessel shadowing. | 0 (No) or 1 (Yes) | 1 |
| analyse\_choroid | Flag to allow OCT choroid analysis. | 0 (No) or 1 (Yes) | 1 |
| analyse\_slo | Flag to allow SLO image analysis. | 0 (No) or 1 (Yes) | 0 |
| custom\_maps | List of custom thickness maps to compute | Comma-separated list of valid retinal layer key pairs, with the custom map inner and outer layer written as “{layer1}\_{layer2}”, i.e. “ILM\_OPL, ONL\_RPE”. The whole retina **will always** be computed regardless of this parameter. By default, the inner and outer retina maps will be measured if available. Write 0 if no custom maps to compute. | 0 |
| analyse\_all\_maps | Flag to compute all individual retinal layer thickness maps, if such layers are available in the metadata. | 0 (No) or 1 (Yes) | 0 |
| analyse\_square\_grid | Flag to measure thickness maps using an 8x8 square grid approximately 7 x 7 mm. By default, the ETDRS grid will be measured for posterior pole OCT scans. | 0 (No) or 1 (Yes) | 0 |
| choroid\_measure\_type | Whether to make choroid measurements according to choroid-axis (perpendicular) or image-axis (vertical). | perpendicular or vertical | perpendicular |
| Linescan\_roi\_distance | Distance temporal and nasal from fovea (on B-scan) to define fovea-centred region of interest. | An integer in interval [500, 3000] | 3000 |

**2) RUNNING OCTolyzer ON YOUR DATA**

1. Launch the Anaconda prompt application (see below)
   1. Select the Windows icon.
   2. Click the drop-down list of the Anaconda3 (64-bit) folder.
   3. Select the Anaconda Prompt (Anaconda3) application.
2. Activate the python environment which stores all python packages necessary to run OCTolyzer.
   1. Type conda activate oct-analysis
   2. Press Enter.
3. Navigate to the OCTolyzer\ directory
4. Run OCTolyzer.
   1. Type python octolyzer\main.py
   2. Press Enter.

**3) OUTPUT FILES FROM RUNNING OCTOLYZER AT THE TERMINAL**

* In the output\_directory there will be a log file analysis\_log.txt storing information printed out to user for each image analysed.
  + **Note:** This can be useful to inspect why an image failed to be analysed, or if the fovea was not detected, etc.
* In the output\_directory there will be a .xlsx file analysis\_output.xlsx storing metadata and feature measurements for each image file row-wise. The file is what should be used for downstream data analysis after batch processing.
  + The first sheet stores all metadata for this file. If there is an empty row with only a filename, this image failed to be analysed.
  + There are separate sheets for OCT scan results, dependent on the type of scan. These comprise a total of three possible .xlsx sheets for OCT B-scans:
    - **Fovea-centred, single line-scans:** features measured are subfoveal thickness, area measured using a 2mm, fovea-centred region of interest. Choroid vascular index is also measured for the choroid.
    - **Posterior pole, macula-centred volumetric scans**: features measured are subregional, average thickness measurements following the standard ETDRS grid (1mm, 3mm and 6mm ROIs) for the retina and choroid (and any inner retinal layers specified in the config.txt file). Volumes are also measured for these regions, as well as choroid vascular index and vessel density for the choroid.
    - **Optic disc-centred, peripapillary scans:** We compute average thickness measurements of the peripapillary choroid, RNFL and whole retina across the peripapillary circular grids, which is centred according to the orientation of the optic disc and fovea.
  + A final sheet stores all the SLO feature measurements called “SLO\_measurements”. This stores commonly measured en face retinal vessel features such as fractal dimension, tortuosity, vessel width, vessel perfusion density and central retinal artery/vein equivalent (including arteriole to venule ratio)
* By default, two additional directories output\_directory/slo\_segmentations and output\_directory/oct\_segmentations are saved out containing images of the segmentation masks superimposed onto the SLO and OCT image files for quick inspection.
  + **Note:** This is helpful for quickly checking the outputs of the OCT and SLO segmentation models.
  + In slo\_segmentations, each file saves out an image of the SLO images stitched together with binary vessels in green, and arteries and veins in red and blue, respectively. The fovea is also plotted. For disc-centred SLO, the B and C zones are plotted, as well as the optic disc centre.
  + In oct\_segmentations:
    - single line-scan analyses save out the B-scan with retinal layer and choroid segmentations
    - posterior pole volumetric scans save out the thickness maps superimposed onto the SLO with the ETDRS grid measurements ontop. This is for the whole retina, choroid and choroid vascular index.
    - Peripapillary scans save out the peripapillary OCT with retinal layer and choroid layer segmentation.
* By default, in output\_directory folders will be created for each image file (using the image’s filename for reference). Stored in each folder are
  + Feature measurements and metadata saved as output\_directory/filename/{filename}\_output.xlsx
  + A .txt file saved as output\_directory/filename/{filename}\_log.txt

This can be helpful if an unexpected error crashes the analysis run halfway through a large batch, as the software will be able to automatically identify previously analysed images based on the existence of output\_directory/filename/{filename}\_output.xlsx

**Delete this file if you wish to re-analyse the file, i.e. when you run the software again it will re-analyse a file when it cannot detect** {filename}\_output.xlsx in **the** output\_directory/filename **folder.**

* In config.txt, by default the option save\_individual\_segmentations is set to 1 will save out the binary segmentation masks, original SLO image and images of the segmentations superimposed onto the SLO.
  + **Note:** You may want to set this to 0 if processing large batches of images, to save on memory consumption. However, the segmentation masks will not be accessible if set to 0, so if memory is not a problem/if analysing a smaller batch, keep this as 1.
  + Moreover, saving these segmentation masks out allows you the opportunity to manually annotate and fix any errors made by the segmentation models.

**4) RUNNING OCTOLYZER USING AN IDE**

* If running interactively from an IDE (VSCode, Jupyter notebooks/lab), use the notebook “usage.ipynb” to get started on using the analyse.py script for your own image file individually.

**5) CORRECTING SEGMENTATION ERRORS**

* We do not have any automatic functionality within OCTolyzer to correct any segmentation errors, and we rely on the user to identify any visible problems with vessel classification, etc.
* We do provide functionality to correct en face retinal vessel and optic disc segmentation via **ITK-Snap**. There are instructions on using ITK-Snap for manual annotations in the directory OCTolyzer/instructions/SLO\_manual\_annotations which describe how to use ITK-Snap and correct the binary vessel mask, and the artery-vein-optic disc segmentation masks.
* Once the segmentation masks are saved out in the same folder with the original segmentation mask which was corrected, the pipeline can be run again and OCTolyzer should automatically identify this additional manual annotation and re-compute the features!