AOSLO Pipeline

Base directory of the code of the AOLSO pipeline (UML diagram that can be found in this file's folder (*aoslo_pipelin/src/classes.png*), generated automatically by running `pyreverse -o png .` in this same folder).

Installing the Anaconda environment

Make sure you have Anaconda installed (if not download it here). Open the *Anaconda Prompt* and go to this specific folder from the pipeline's base directory (if when opening anaconda prompt you end up in P: folder, type `C:` to go in computer's C disk):

```
cd src
conda env create -f env.yml
```

Then you can activate the environment named **aoslo** by typing: `conda activate aoslo`

If the installation fails, you might need to follow the instructions in *T:\Studies_tutorials\anaconda\conda_setup.txt* and in *T:\Studies_tutorials\anaconda\pip_install_with_hojg_connection.txt*.

If the error persists, you can also create first the environment from the txt file:

```
conda create --name aoslo --file requirements.txt
```

Then activate the environment named **aoslo** by typing: `conda activate aoslo`

If there are still errors, you can install packages one by one y hand with conda first and if not work with pip.

Code structure

The code is structured in the following way:

1. AST: Graphical User Interface to help doctors label the vessels in the fundus images (AST = Aquisition Support Tool). More informations are available on the *README* file located in the *AST* folder.
2. cell: Cell detection and analysis algorithms, including cone detection methods, but also cone densities. More informations are available on the *README* file located in the *cell* folder. To run this code, go in the cell folder (`cd cell`).
3. **configs**: Configuration files to run the pipeline with various parameter settings.
4. **Helpers**: Shared utility functions used by multiple components of the pipeline. Normally old + deprecated functions, to be removed.
5. input_pipeline: Input data processing pipeline for automated AOSLO videos to images registration. In here lies the code that is used in the Virtual Machine that is installed on the computer in the AOSLO

room and that can also be accessible via distance from each computer (Connexion Bureau à distance) with the account *ADI\aoslo* (ask Mattia for password). The code is used once the AOSLO videos have been taken and its purpose is to automate the process of using the BostonMicromachines (BMC) software in order to get the different AOSLO modality images. More informations are available on the *README* file located in the *input_pipeline* folder. To run this code, go in the input_pipeline folder (`cd input_pipeline`).

6. **pdf_report**: PDF report generation utilities.
7. **plotter**: Visualization and plotting utilities for analysis results.
8. **readme_images**: Image files used in documentation and README files.
9. **shared**: Shared resources and common components across different modules.

For developers, use Visual Studio Code for code modifications and project management.

AOSLO Pipeline

Handles all the postprocessing code for analyze AOSLO images

Introduction

How to run the pipeline

If you need to install Anaconda and create your environment to code for the first time, see here.

The pipeline can either be run via the GUI from this directory go into *input_pipeline* (`cd input_pipeline`) and run `python main.py` (see README in *InputGUI2* folder for more information) or directly from a python terminal by launching the following command in a correct environment and with the configuration file (*configs/config.txt*, more informations down below for config options) set up:

```
python Start_PostProc_Pipe.py
```

Required Set-Up in the computer

To run all the steps you will need to install/set-up following:

- Matlab Mathworks Runtime version **R2022a (9.12)** download for the Montaging step (**WARNING** you need admin rights). Once download, add the path to the downloaded Matlab Mathworks to the system's environment variable (press Windows button, type *Modifier les variables d'environnement système*, press *Varibales d'environnement..*, on *Path* click *Modify* and add a new variable: *C:\Program Files\MATLAB\MATLAB Runtime\v912\runtime\win64*) and restarts the computer.
- Matlab's package for Montaging: go to (cd)*Montaging\AOAutomontaging_master\AutoAOMontaging\for_redistribution_files_only* and type `python setup.py install` to install Matlab's package for Montaging (more details in the Montaging step)

Steps

alt text

The pipeline works only on images. The following steps are:

If working on cone images:

1. Montaging of individual images (Montaging)

If Density Analyis: 3. Cone Density computation (Density Analysis) 4. Comparison with OCT Layer Thicknesses (Layer Thickness) 5. All subjects densities analysis (Global Density Analysis)

WARNING When running Density script be sure Montaging and Cell detection (ATMS and MDRNN) steps are done. For global density and global diabetic, subjects need to have their density or diabetic analysis respectively computed.

Files:

- save_layer_thickness.py: Get the layer thicknesses from OCTs.
- save_layer_features.ipynb: Get the layers features from OCTs.

Config options

- **[os] __base_dir** = base directory where the patients' data lies. It can be either a single subject's folder or the directory containing all the subjects' folder as subfolders

- **[os] __powerShell** = complete path of the location of the powershell executable

- **[Montaging] __do_montaging** = wheter or not we want to do the montaging step.

- **[Montaging] __GUI_done** = whether the Montaging GUI has already been done. If yes it will run the subsequent step of Montaging and/or the subsequent pipeline steps. If not, it will run Matlab's code for montaging.

- **[Montaging] __GUI_location_file** = the name of the Montaging GUI output file to search to reconstruct the fundus eye image.

- **[Montaging] __do_SSIM** = whether or not we want to apply SSIM correction on the results from the Montage GUI

- **[Montaging] __SSIM_n** = if applying SSIM correction, number of pixels in each direction to look for best overlap between overlapping components.

- **[Montaging] __parallel** = whether or not to run the SSIM correction in parallel (if you're not using your computer) or not

- **[Montaging] __montaged_dir** = the name of the montage directory

- **[Montaging] __corrected_dir** = the name of the montage corrected directory (after having apply the GUI correction and SSIM if run)

- **[Montaging] __ssim_dir** = the name of the SSIM directory for intermediary results

- **[Registring] __do_registring** = whether or not we want to do the registring step (for Diabetic analysis on OA850nm images)
- **[Registring] __do_analysis** = whether we want to do the full analysis, or only apply POC (best registration algorithm)
- **[Registring] __register_dir** = the name of the register directory where results of the step will be found
- **[Registring] __csv_name** = the name of the CSV with the results
- **[CellDetection] __do_mdrnn** = Whether or not we want to run the MDRNN CellDetection
- **[CellDetection] __do_analysis** = whether we want to run the full MDRNN analysis (test for the best preprocessing method to detect the cones)
- **[CellDetection] __methods** = methods to apply when doing the MDRNN analysis
- **[CellDetection] __replaces** = replacement methods to apply when doing the MDRNN analysis
- **[CellDetection] __range_methods** = range of methods to apply when doing the MDRNN analysis
- **[CellDetection] __specials** = special methods to apply when doing the MDRNN analysis
- **[CellDetection] __corrects** = correcting methods to apply when doing the MDRNN analysis
- **[CellDetection] __treats** = treating methods to apply when doing the MDRNN analysis
- **[CellDetection] __visualize_patches** = whetehr or not to visualize patches doing the MDRNN analysis
- **[CellDetection] __mdrnn_overlap_distance_same_image** = the maximum overlap in pixel between cones in a single CalculatedSplit image
- **[CellDetection] __mdrnn_overlap_distance** = the maximum overlap in pixel between cones in overlapping CalculatedSplit images
- **[CellDetection] __do_atms** = whether we want to apply ATMS Cell Detection
- **[CellDetection] __atms_overlap_distance** = the maximum overlap in pixel between cones in overlapping Confocal images
- **[CellDetection] __output_dir_atms** = the output directory name of ATMS CellDetection
- **[CellDetection] __axial_length_file** = the name of the file qith subject's axial length
- **[Diabete] __do_diabetic_analysis** = whether to do Diabetic Biomarker analysis
- **[Diabete] __diabetic_analysis_dir** = directory name of the output of diabetic analysis
- **[Diabete] __do_global_analysis** = whether or not to do the global diabetic analysis, once diabetic analysis has been done for all subjects
- **[Diabete] __global_diabetic_dir** = directory name of the output of global diabetic analysis
- **[Density] __do_density_analysis** = whether to do Cone Density analysis

- **[Density] __growing_factor** = the linear function to include cones or layer thicknesses values in the density computation
- **[Density] __center_limit** = the limit to which we do not trust on ATMS CellDetection algorithm
- **[Density] __curve_fit_limit** = the outer limit to which we approximate the logarithmic function in the center of the eye
- **[Density] __atms_limit** = the higher limit of ATMS agorithm in eccentricity
- **[Density] __mdrnn_limit** = the lower limit of MDRNN agorithm in eccentricity
- **[Density] __analysis_dir** = the name of the density analysis direction for outputs
- **[Density] __do_global_analysis** = whether or not to do the global density analysis, once density analysis has been done for all subjects
- **[Density] __global_density_dir** = directory name of the output of global density analysis
- **[LayerThickness] __do_compare_to_layer_thickness** = whether or not to compare densities with layer thicknesses from OCTs
- **[LayerThickness] __layer_thickness_dir** = directory name of the output of layer thickness analysis
- **[BloodFlow] __do_blood_flow** = whether or not we want to do the blood flow analysis
- **[BloodFlow] __blood_speed_estimation** = whether or not we want to do the blood speed estimation (single subjects, done already if select *__blood_model*)
- **[BloodFlow] __blood_model** = whether or not we want to do the blood flow model analysis (multiple subjects)
- **[BloodFlow] __vessel_size** = sized of the vessel we want to analyze
- **[BloodFlow] __model_output_dir** = directory name of the output of the blood model analysis
- **[BloodFlow] __speed_output_dir** = directory name of the output of the blood speed analysis