Evaluation of the FCN

All data needed for the evaluation can be found in the "Benchmarks" folder. The data and associated plots were generated using MATLAB. Each experiment therefore has a folder called "MATLAB". All used scripts can be found in this folder. Their name should be self-explanatory. The codebase for these Experiments can be found here:

- Dependence of the FCN on the degree of occlusion
- Dependence of the FCN on the degree of occlusion and pitch rotation
- Dependence of the FCN on the degree of occlusion and roll rotation
- Dependence of the FCN on the degree of occlusion and yaw rotation
- Dependence of the FCN on the pitch rotation
- Dependence of the FCN on the roll rotation
- Dependence of the FCN on the yaw rotation
- Dependence of the FCN on the pitch and roll rotation
- Dependence of the FCN on the pitch and yaw rotation
- Dependence of the FCN on the roll and yaw rotation

The naming of the used input data follows the following pattern: <id>_ <ImgNo>$<yaw>_<pitch>_<roll>.png. With the parametric face image generator that can be downloaded here, new data can be generated. For example, with a different type of occlusion. The software comes with a detailed description on how to choose between different occlusion types.

Fitting with Real-Life Data

1. Clone the fitting script from `https://github.com/Arneli/bsc-integratingcnnsegmentation`.
2. Clone the data container from `https://github.com/Arneli/image-data`.
3. Replace the `data_in` folder of the fitting script with the `real_life_images/data_in` folder of the data container.
4. Since there are no ground truth rps files, delete the `PARAMETRIC_rps_files` folder of the fitting script.
5. Delete the `dummy` file inside the `data_out`, the `fits`, the `rps`, and the `segmentations` folder of the fitting script.
6. Copy the 'face12' version of the Basel face Model into the `bfm` folder of the fitting script.
7. Update the for-loop in `src/main/scala/FitWithOcclusions.scala` of the fitting script that it runs over all images in `data_in`. Replace line 50 of `src/main/scala/FitWithOcclusions.scala` with `for(i <- 0 to 11) {`.
8. Run `sbt assembly` inside the fitting script folder to create a `target` folder.
9. Run `java -cp target/scala-2.12/bsc-integratingCNNSegmentation.jar FitWithOcclusions -d -m bfm/model2017-1_face12_nomouth.h5 -n 1000 -f DUMMY` to create fits without any mask.
10. Run the above command again and replace `DUMMY` with `EGGER` or `FCN` to create fits with the mask of the top-down approach of Egger et al, or the segmentation of the FCN.

Fitting with Synthetic Data

1. Produce the fits

1. Clone the fitting script from `https://github.com/Arneli/bsc-integratingcnnsegmentation`.
2. Clone the data container from `https://github.com/Arneli/image-data`.
3. Choose an **occlusion-type**. The choices are: Colored_boxes, hands, glasses, micros and no_occlusion.
4. Choose a experiment **setting**. The choices are:
   - **setting1_(face12_50parameters)**: The tailored version of the Basel Face Model is used for both the ground truth mask and the rendering.
   - **setting2_(bfm_50parameters)**: The rendering uses the original version of the Basel Face Model which shows more skin than only the face. For the ground truth mask, the tailored version is used.
5. Replace the `data_in` folder and the `PARAMETRIC_rps_files` folder of the fitting script with the `<occlusion-type>/<setting>/data_in` folder and the `<occlusion-type>/<setting>/PARAMETRIC_rps_files` folder of the data container.
6. Delete the `dummy` file inside the `data_out`, the `fits`, the `rps`, and the `segmentations` folder of the fitting script.
7. Copy the 'face12' version of the Basel face Model into the `bfm` folder of the fitting script.
8. Update the for-loop in `src/main/scala/FitWithOcclusions.scala` of the fitting script that it runs over all images in `data_in`. Replace line 50 of `src/main/scala/FitWithOcclusions.scala` with `for(i <- 0 to 9) {`.
9. Run `sbt assembly` inside the fitting script folder to create a `target` folder.
10. Run `java -cp target/scala-2.12/bsc-integratingCNNSegmentation.jar FitWithOcclusions -d -m bfm/model2017-1_face12_nomouth.h5 -n 1000 -f DUMMY` to create fits without any mask.
11. Run the above command again and replace `DUMMY` with `EGGER`, `FCN`, or `GROTRU` to create fits with the mask of the top-down approach of Egger et al, the segmentation of the FCN, or the ground truth mask.

2. Produce the Errorplots

1. Clone the evaluation script from `https://github.com/Arneli/bsc-MATLAB-evaluation-script`.
2. Copy the outputs of the fitting script into the subfolders of the evaluation script:
   - Copy the contents of the `data_in` folder of the fitting script into the `data_in` folder of the evaluation script.
   - Copy the contents of the `data_out` folder of the fitting script into the `data_out` folder of the evaluation script.
   - Copy the contents of the `PARAMETRIC_rps_files` folder of the fitting script into the `PARAMETRIC_rps_files` folder of the evaluation script.
   - Copy the contents of the `rps` folder of the fitting script into the `RPS` folder of the evaluation script.
   - Create an empty folder named `image_data` inside the folder of the evaluation script.
   - Create a file named `posteriors.txt` inside the `posteriors` folder of the evaluation script. Then fill this file with the content from all the posterior files of the fitting script (e.g. posteriors_EGGER.txt).
3. Run `posteriors/read.m` to generate `image_data/posterior_matrix.mat`.
4. Change the for-loop on line 4 of `generate_image_data_6_to_50.m` and `generate_image_data_first5.m` to run over all '.rps' files.
5. Run `generate_image_data_6_to_50.m` and `generate_image_data_first5.m`. That should produce two additional files inside the `image_data` folder.
6. Update the for-loop in the MATLAB-Script `plot_image_data_average.m` on line 4 to run over all data files inside the `image_data` folder. Then run it to produce the errorplots.
7. Run `segmentation_fit_iteration.m` to plot the changes of the error during the iterations visually.