Project 2: More Generative Models

- ▷ Projects should be submitted in teams of 3-4 students.
- ▷ Accepted format is a single zip file of your **Python scripts/notebooks** containing your results and a report (**2 A4 pages** - font size 12). Please include the full names and student ID numbers of all team members in the report.
- ▷ Submissions are made via a Sciebo dropoff link: `https://fz-juelich.sciebo.de/s/8fjSCb96KcJVFd9`.
- ▷ Submissions have to follow the naming scheme *project02_USER1_USER2_USER3_USER4.zip* using your university username ("Benutzername" in ILIAS/IDM). When a second version is uploaded use the ending *_v2.zip* for the file. Only the latest version will be rated. Make sure you name the zipped folder the same as the zip file.
- ▷ The project is either a success or a failure.

The brain contains numerous nerve fibers that connect individual neurons and allow the transmission of information through electrical impulses. Three-dimensional polarized light imaging (3D-PLI) is a microscopic imaging technique designed to visualize nerve fibers and their 3D orientations in brain tissue sections. It measures several physical properties of the tissue, including birefringence and light attenuation. Measurements are recorded in distinct parameter maps.

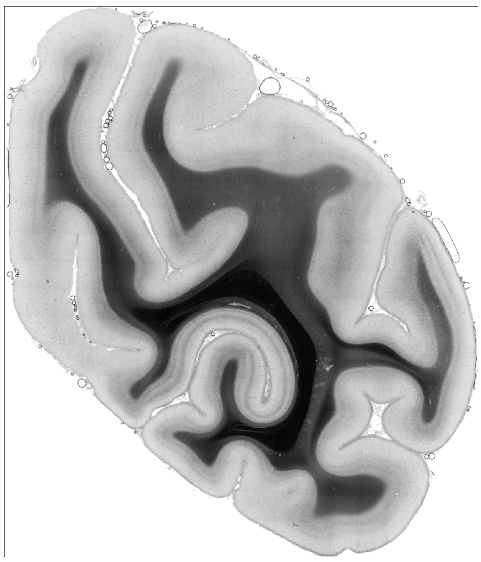


Figure 1: Transmittance map

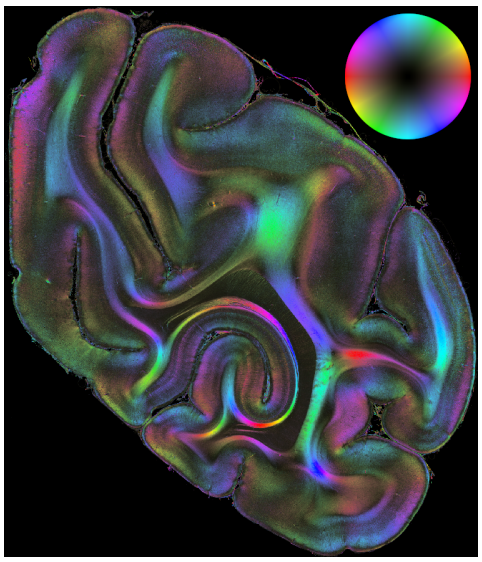


Figure 2: Fiber orientation map (FOM).

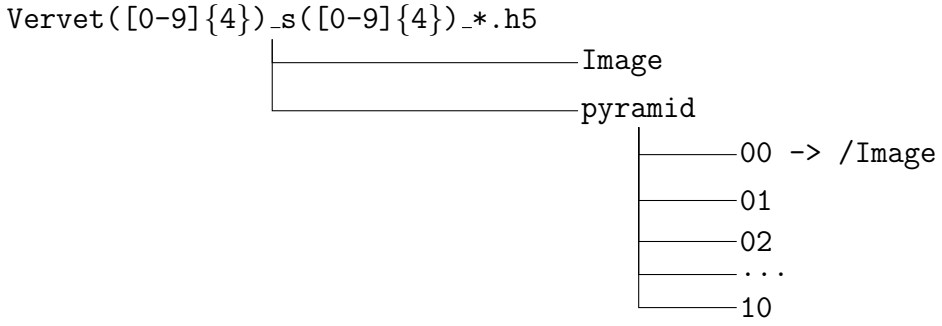
The *transmittance map* (Fig. 1) mainly shows the attenuation of light passing through the tissue. It therefore resembles a shadow of the tissue, being darker in areas with densely packed nerve fibers and lighter elsewhere. The maps are stored as gray-scale images.

The *fiber orientation map (FOM)* (Fig. 2) visualizes the 3D orientation of nerve fibers at each pixel. It utilizes the HSV-colorspace to encode in-plane direction by the hue and out-of-plane inclination by the value and saturation (darker areas mean more out-of-plane fibers). The maps are stored as RGB color images.

1. **Data Exploration**

   Under the following Sciebo link you will find brain sections of two vervet monkey brains measured with 3D-PLI: https://fz-juelich.sciebo.de/s/F7jJcSj1le1pFeY. While Vervet1818 was sectioned in the coronal plane (from the front to the back), Vervet1947 was sectioned in the sagittal plane (from one side to the other). You will use Vervet1818 for training and Vervet1947 for evaluation. For each brain section, you will find one transmittance map and one FOM in the respective folders (`transmittance/` and `fom/`).

   The files are stored in HDF5 format and quite large (up to 6.2 GB). If you are not familiar with the HDF5 format, check out the h5py documentation for more information. Each section file has the following structure:



   The first number in the filename shows the ID of the vervet monkey and the second number is the section ID. For some section IDs there are multiple files for different regions (e.g. left and right hemispheres).

   In each image, the HDF5 dataset `Image` contains the full resolution parameter map (1.3 µm resolution) of shape `(height, width, 3)` for the FOMs with 3 RGB color channels and `(height, width)` for the transmittance maps. All pixel values are stored as unsigned 8-bit integers in the range $0 \dots 255$.

   The HDF5 group `pyramid` consists of datasets `00` ... `10`. Each of these datasets is a downsampled version of the original image by a factor of $2^{\mathtt{n}}$, where `n` is the pyramid number. As dataset `00` stores the original resolution, it is simply a link to the `Image` dataset. The pyramid structure allows us to visualize the images at different zoom levels without loading the complete section.

   To get familiar with the dataset perform the following tasks:

   (a) Download some sections of *Vervet1818* and at least one of the *Vervet1947* sections.

   (b) Visualize an appropriate pyramid level for each of the downloaded sections (FOM and transmittance) to get an impression of the full content.

   (c) Locate some interesting structures and visualize full-resolution crops in the FOM and transmittance maps. Do not load the full section into memory for this.

2. **Sampling Strategy**

   We will first have a look at the FOMs of Vervet1818. Since all sections probably won't fit into your memory, we need to sample smaller image patches from disk for training. Write a PyTorch DataLoader and Sampler to load small image patches from the files. You can follow the following steps for this:

(a) For the **DataLoader** you can adapt your implementation from the first project. It should store an open h5py File handle for each of the files you want to access (opening and closing each time would cost much time). Adjust the `__getitem__()` method such that it takes a tuple of

(`brain`, `section`, `region`, `map_type`, `row`, `column`, `patch_size`)

as input and loads a patch of size (`channels`, `patch_size`, `patch_size`) with full image resolution from location (`row`, `column`) in `brain` (Vervet1818 or Vervet1947), `section` (one of the available section numbers), `region` (left, right, cerebellum or none) and the parameter map of `map_type` (FOM or transmittance). The attribute `channels` depends on the requested parameter map. Convert the patch to a `torch.Tensor` of type `float32` and normalize the pixel values to the range $[0, 1]$.

(b) You can adapt your **Sampler** from project 1 such that it fits the new data scheme. Overwrite the `__iter__()` method such that it yields random tuples of

(`brain`, `section`, `region`, `map_type`, `row`, `column`, `patch_size`).

Make sure that it only provides locations for patches that will fit completely into the image (i.e. consider the `patch_size` when sampling).

3. **Image Generation**

Now you have everything ready to implement a generative model of your choice on sampled FOM patches. We exclude a VAE and variants from the choice since it was already part of project 1. For the generative model, you can choose any type (some of them will appear later in the lecture):

- GANs
- Autoregressive models
- Flow-based models
- Diffusion Models
- ...

Some recommend examples of specific models would be a Wasserstein GAN, StyleGAN, PixelRNN, PixelCNN, Glow or Stable Diffusion. If you have no preference, we suggest starting with a GAN model, as they produce pleasing images without requiring extensive computational resources. You can start with a patch size of 64 pixels and, depending on your chosen model and hardware, might experiment with larger or smaller sizes.

(a) Train the model of your choice on the downloaded *FOM* maps of *Vervet1818*.

(b) Generate some samples with your model and visualize them.

4. **Image-to-Image Translation**

3D-PLI provides different parameter maps for the same brain section. As an example, you have learned about the transmittance and FOM in the introduction. An interesting

question is to what extent the information contained in one map can be used to generate the other.

In this task, you will implement a generative model of FOM maps conditioned on corresponding transmittance maps to address this question. Such models perform an image-to-image translation. Again, you are free to implement any type of generative model you like. Examples of conditional generative models that can perform an image-to-image translation are a Conditional GAN, CycleGAN or Pix2Pix. If you have no preference, we suggest using the Pix2Pix model.

Again, you will need a sampling strategy to train the model with smaller image patches. You need to adjust your data sampling a bit for this task. Adjust your **DataLoader** to accept a sampling location without `map_type`, such as

$$(\texttt{brain},\ \texttt{section},\ \texttt{region},\ \texttt{row},\ \texttt{column}, \texttt{patch\_size})$$

and adjust the `__getitem__()` function to return a tuple (or dictionary) of two patches. One is the transmittance and the other is the corresponding FOM from the same location.

(a) Train the image-to-image model of your choice to produce a FOM given a transmittance patch.

(b) Generate FOMs for transmittance patches at various locations in the Vervet1947 brain and visualize them along with their transmittance input and target FOM.

5. **Evaluation**

In this task, you will assess the visual appearance and accuracy of your generated samples. Use only data from the Vervet1947 brain for the evaluation.

(a) Compute a Fréchet Inception Distance (FID) score for 1000 samples of your generative model from task 3. Compare your generated samples with 1000 real samples of Vervet1947.

(b) For task 4, you have access to target images (the real FOMs) for each generated sample. Implement the following pixel-level metrics (or use a library such as scikit-image): Mean Squared Error (MSE), Structural Similarity Index Measure (SSIM), and Peak Signal-to-Noise Ratio (PSNR). Compare 1000 generated images with their ground truth and report the overall mean and standard deviation of scores.