Applying the VoxelMorph Framework to C. Elegans Brain Data

Gasser ELBANNA*, Louise PLACIDET*, William VERSTRAETEN*,Sahand Jamal RAHI †
*École Polytechnique Fédérale de Lausanne (EPFL)
†Laboratory of the Physics of Biological Systems, Institute of Physics, EFPL

I. Introduction

Manually labeling the regions of interest for multiple images of biological samples can be ineffective and time-consuming. Solving this issue requires either implementing image processing algorithms or learning-based approaches. Learning-based approaches offer faster and computationally more efficient registration [1].

With the VoxelMorph framework, it is possible to register all the frames (named, moving frames) of a recording to a single fixed frame and learn how the moving frames can be warped onto the fixed (*i.e* compute the deformation fields). These deformation fields can warp the manually annotated regions of interest of the fixed frame onto the moving frames. The purpose of this paper was to find these deformation fields, using the VoxelMorph framework.

More specifically, the data we used comprised 3D volumes of a single *C.Elegans* moving over time. Each frame consisted of a 3D-scan of the *C.Elegans* obtained through confocal microscopy, revealing fluorescently labeled neurons of the subject in only certain z-stacks. Moreover, some of these volumes have corresponding masks of the neurons which were manually annotated: figure 1 shows a max intensity projection through the z-dimension of such a volume and its corresponding mask.

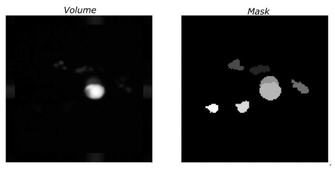

Figure 1: Example volume of the *C.Elegans* and its corresponding hand annotated mask.

II. Related Work

Extensive work exists in 3D medical image registration, providing several methods to compute the deformation fields for volume sets. Some methods seek to optimize the model within the space of displacement vector fields, such as Large Diffeomorphic Distance Metric Mapping and standard symmetric normalization (SyN) [2, 3]. Other 2D registration methods, such Optical flow estimation, perform registrations and return a displacement vector field for each pair of 2D images [4]. However, these traditional registration methods optimize the objective function for each pair of images, resulting in substantial computations and sometimes limited deformation fields [1].

VoxelMorph extends this work, by providing a framework that learns a function which maps an input image pair to a deformation field. This framework optimizes parameters of a neural network on a *set of images* instead of each pair. Not only is this framework's accuracy comparable to the state of the art, but it considerably speeds up the registration process (from several hours to minutes on a CPU and under a second on a GPU) [1]. As a result, the VoxelMorph framework was implemented to obtain the desired deformation fields obtained from the numerous 3D volumes.

III. The VoxelMorph Framework

A. The Framework

This framework takes in a pair of volumes, respectively called "fixed" (f) and "moving" (m). With these training volumes, the U-Net learns to find the optimal network parameters theta for the function:

$$g_\theta(f, m) = u \tag{1}$$

This results in the registration field Φ, defined as

$$\Phi = Identity + u \tag{2}$$

[1]. This deformation field is then used to warp the moving volume onto the fixed volume. This warped moving volume is referred to as the moved volume. Since the moved is an approximation of the fixed volume: we then estimate how close the deformation of the moving volume is to the fixed.

We originally used two different losses to optimize the U-Net architecture. The first is the mean squared error (MSE) of the intensity: comparing the intensity difference between the voxels of our predicted volume (moved), and the voxels of the fixed volume. The second loss which we used is referred to as the smooth loss, which corresponds to a diffusion regularizer on the spatial gradients of the displacement u, enforcing smooth changes in the deformation field. As a result, when training the unsupervised model the objective function considered is the following:

$$\begin{aligned} \boldsymbol{L}_{unsupervised}(f, m, \phi) = \boldsymbol{L}_{similarity}(f, mo\phi) \\ + \lambda \boldsymbol{L}_{smooth}(\phi) \end{aligned} \tag{3}$$

Moreover, segmentation can be leveraged as auxiliary information to optimize this model. By utilizing the 3D (manually annotated) masks associated with specific volumes, a moving mask can be warped onto another fixed mask, using the computed deformation field. Based on the overlap of the predicted mask and the fixed, it is possible to compute the dice score, as summarized in figure 2. This defines a new objective function:

$$\begin{aligned}\boldsymbol{L}_{semi-supervised}(f, m, \phi, s_f, s_m) = \\ \boldsymbol{L}_{similarity}(f, mo\phi) + \lambda \boldsymbol{L}_{smooth}(\phi) + \gamma \boldsymbol{L}_{seg}(s_f, s_m o\phi)\end{aligned} \tag{4}$$

with s_f and s_m defined as the fixed and moved masks respectively.

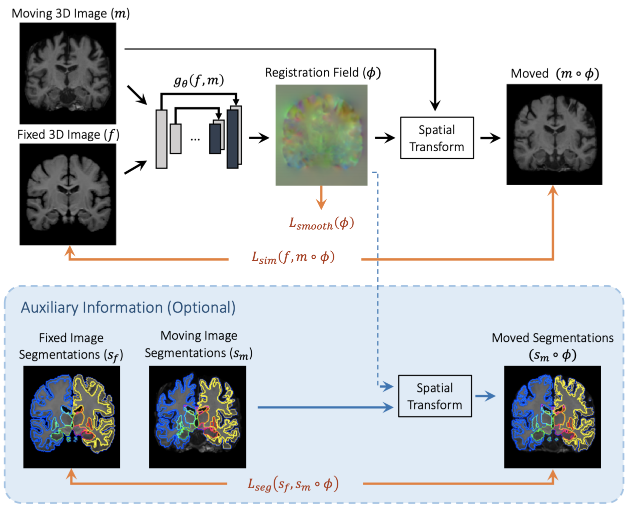
Moving 3D Image (m)
Fixed 3D Image (f)
$g_\theta(f,m)$
Registration Field (ϕ)
Spatial Transform
Moved ($m \circ \phi$)
$L_{smooth}(\phi)$
$L_{sim}(f, m \circ \phi)$
Auxiliary Information (Optional)
Fixed Image Segmentations (s_f)
Moving Image Segmentations (s_m)
Spatial Transform
Moved Segmentations ($s_m \circ \phi$)
$L_{seg}(s_f, s_m \circ \phi)$

Figure 2: The VoxelMorph Architecture, diagram from [**VoxelMorph Framework**]. The framework takes in 2 volumes and outputs a deformation field. This field is applied onto the moving volume to obtain a prediction, which is then compared to the fixed. The framework can also use auxiliary data during the training, shown in the blue section.

B. The C.Elegans Data

The dataset, provided by the Laboratory of the Physics of Biological Systems at EPFL, was in Hierarchical Data Format (HDF). The first step consisted in extracting the data which comprised 1715 3D-volumes of worm images, each corresponding to a specific time frame of the given recording. Moreover, 118 of these volumes had corresponding masks manually segmented, which we leveraged as auxiliary data for our model. Figure 3 illustrates how the dataset is stored in the HDF format.

The provided volumes were obtained after applying image processing algorithms concerning image registration: the first step included thresholding the intensity and only keeping the three brightest objects (neurons). This was followed by the implementation of a non-rigid deformation method (proposed by Jian and Vemuri [5]): these three objects were filled with random points. Finally, affine transformations are applied to

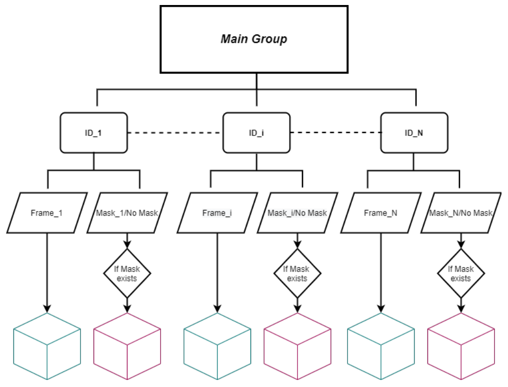

Figure 3: Diagram explaining the dataset format and content. Each volume represents a time frame in 3D and some volumes have corresponding 3D masks.

align these three objects. The final volumes area obtained after cropping around these regions of interest. This is considered to be our baseline method to compare with the VoxelMorph framework.

C. Implementation

These input volumes were pre-processed to be adapted for the framework. They were first flipped to be in a (z, x, y) format (instead of the original (x,y,z) format). In order to have comparable intensity scales inter-volumes, they were also normalized. Moreover, the volumes were padded along the x and y directions (to be 128x128), given that the models with the highest performance have inputs that are sized as multiples of 2^N [6]. However, the framework used constant padding, which was not adapted to this data (see Figure 4).

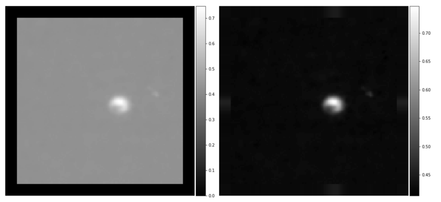

Figure 4: (left) is when applying constant padding, (right) is when applying mean padding.

As a result, the padding was modified to take the mean, as this better preserved the contrast within the volume. Lastly, these volumes were fed to the model as arrays.

Once these volumes were correctly pre-processed, they were passed to a data generator. Considering the high amount of RAM required when loading multiples volumes simultaneously, the data generator solves this issue, by randomly loading a few volumes at a time from the considered set of volumes. The train generator provided by the framework only generated 2D slices from volumes, which was not adapted for our particular case [6]. Indeed, focusing on only 2D slices would not take into account the possible movements along the z-dimension. As a result, this train generator was modified to randomly select 3D volumes from a given set.

Given the satisfying results from applying the VoxelMorph training in [1], the architecture of the U-Net was preserved for our particular case: four 32-filter convolutional layers for the encoder and five 32-filter convolutional layers in addition to a 16-filter convolutional layer for the decoder, as illustrated in Figure 5 [1].

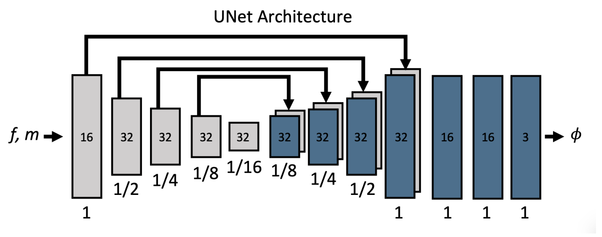

Figure 5: The U-Net Architecture, diagram from [**VoxelMorph Framework**]

IV. Experiments and Results

A. *VoxelMorph unsupervised*

Once the VoxelMorph Framework was functional for the 3D *C. Elegans* dataset, the next step consisted in hyperparameter tuning. To evaluate the performance of the trained models on unseen data normalized cross-correlation (NCC) was implemented. For every model the unseen data consisted of 300 pairs of volumes from the validation set. The models then warped the moving volume to the fixed for every pair considered, producing predictions. The NCC of these predictions and their corresponding fixed volumes were computed to quantify the performance. The last step consisted in averaging the NCCs and setting it as the model's score. The results for the baseline and the best models are presented in table 1.

Furthermore, the unsupervised model performs better on volumes where the affine transformation fails, as seen in figure 6.

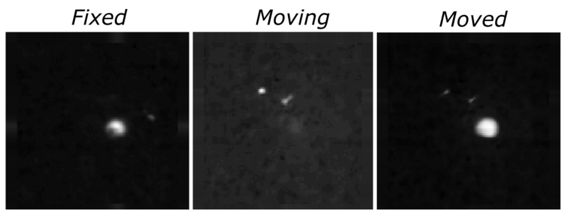

Figure 6: Projection in the z-direction of a moving volume (middle) and the corresponding moved volume (right) after applying VoxeLmorph to allign it to the fixed volume (left). The model used was the one obtained by using a regularizer λ of 0.001 on the flow. NCC score baseline = 0.14 and NCC score unsupervised model = 0.85.

The training was done on 25 epochs with 100 steps and a learning rate of 1e-4 (Adam optimizer), given that these value were deemed sufficient to lead to convergence [6] and in retrospect were indeed enough. Moreover, the VoxelMorph paper studied the effect of the training set size on the accuracy and concluded that when training with 100 volumes or the full dataset, no significant difference was noted [1]. Indeed, in our particular case, the performance of our model when increasing the number of volumes also stagnated at 100 volumes, as shown in figure 7 [1]. The other hyper-parameter was the regularizer enforcing smooth deformation (λ). We tried several values as mentioned in table 1 and it was noted that when shrinking the regularizer, the model performed better. This is probably due to the fact that VoxelMorph is given more freedom to improve the similarity without being restricted by a smooth flow, making it an easier problem to solve.

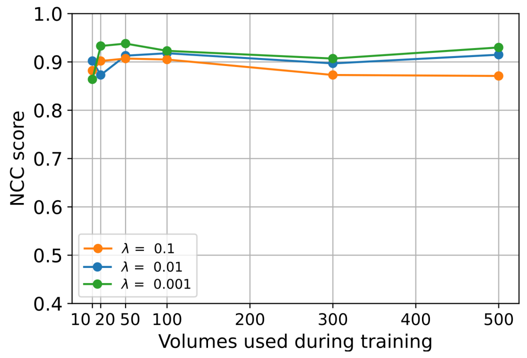

Figure 7: NCC scores of the different trained models on the validation volumes. Models trained with more than 100 volumes barely increase in performance compared to those trained on 100 volumes.

Apart from quantifying the performance of the model some visualisation attempts were made. We first tried to visualize the flow (deformation field) for each time frame (3D volume) as illustrated in figure 8. However, this representation is hard to interpret given the three dimensions. Another attempt consisted in the 3D flow along the three different axes, shown in figure 9.

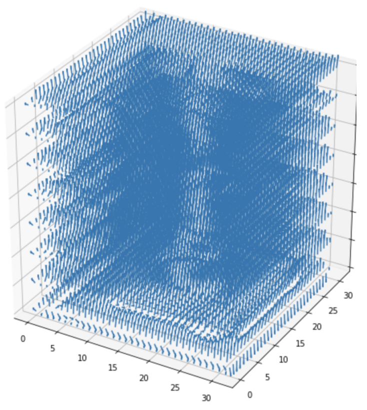

Figure 8: 3D Flow (Deformation Field) of one frame (3D Volume).

Lastly, we tried to obtain a dice score for the unsupervised models (trained without masks). To do so, we fed these models

[1]This justifies why data augmentation was not implemented, given we had access to more than enough volumes.

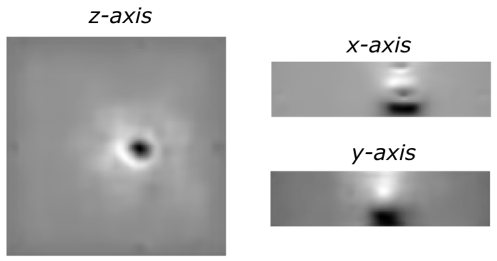

Figure 9: Flow (Deformation Field) of one frame (3D Volume) Projected along the 3 axes. Gray pixels represent a lack of flow in the direction of projection, whereas brighter pixels represent a flow in the direction of the projection and darker pixels show flow in the opposite direction.

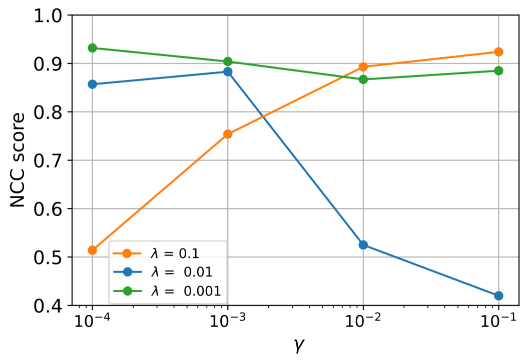

Figure 10: NCC scores of the semi-supervised models on the validation set when varying λ and γ.

the masks instead of their corresponding volumes. As a result, the models tried warping the moving mask as if it was a 3D-scan. We then compared the prediction it made to the fixed mask, by computing the dice score. However, since these models were not trained on masks but on the actual 3D-scans of the *C.Elegans* both the computed dice score and NCC were low (a dice score 0.05 for the unsupervised model against 0.26 for the baseline and a NCC of 0.20 for the model compared to 0.42 for the baseline). These results highlight the sensitivity of the model: it is not able to adapt when given masks instead of the *C.Elegans* volumes it was trained on.

B. New Additions

An addition to improve the model's performance was to take into account the masks (manually annotated regions) associated with some of the volumes during the training. The training of the models and more specifically the train generator was adapted to take into account these masks, and introduced a third score, namely the Dice score for the U-Net to optimize on [2]. With this score was defined a new hyperparameter: the dice score regularizer γ. The volumes and their corresponding masks were then divided into three groups, 70 were used for training, 24 for validation and the last 24 for testing. Different values for both the smoothness (λ) and the dice score (γ) regularizers were tested during the training, the performance of these models on the validation set are resumed in figure 10. Plot 10 reveals no clear trend in results when varying λ and γ. The highest NCC score obtained with a semi-supervised model remained comparable to that of the optimal unsupervised models, the different scores of the models are resumed in table I.

Model	λ	γ	NCC
Affine Transformation (Baseline)	–	–	0.4156
Unsupervised Model	0.01	–	0.9187
Semi-supervised Model	0.001	0.0001	0.9323

Table I: Summary of the NCC scores on unseen data for the best performing supervised and semi-supervised models.

[2]The current version of the library is not adapted to the new version of TensorFlow. As a result, the class had to be redefined, see our notebook.

Another implementation consisted in down-sampling the volumes before training the unsupervised models. Indeed, down-sampling the volumes allowed for much faster training and the use of considerably more volumes (reducing the RAM required). Naturally, there is a trade-off: the higher training speed comes with a loss in resolution. The use of this down-sampling depends on the application: in our case, it was preferred to maintain a higher resolution. Yet, this idea should be taken into account in other applications.

V. Conclusion and Discussion

Once our best model was identified, the last step consisted in computing the deformation field and predicting the moved image for all volumes in a single recording with respect to a unique fixed volume. These deformation fields and predictions were then saved together in an HDF file. Thus, these deformation fields can be used to optimize the labeling process: by labeling a single volume the identified regions of interest are automatically labeled for the rest of the volumes.

The sensitivity of the VoxelMorph should nonetheless be noted. Indeed, the models were trained on volumes with normalized intensities. When testing their performance on unseen and not-normalized volumes, these models performed considerably worse. This was noted in [6] and confirmed in our case. Therefore, to increase the robustness of the model, both normalized and non-normalized volumes could be fed to the train generator. Another addition could be applying data augmentation to these volumes.

For the semi-supervised model, we are considering auxiliary information. With this additional information, one could expect the model to perform considerably better than the unsupervised model. However, this was not observed. This could be due to the limited amount of volumes that were available for the training of the semi-supervised model (at most 70 volumes against 100-500 volumes for the unsupervised).

To conclude, the image processing steps applied to the data were able to take care of the general affine movements, but the VoxelMorph Framework provides an effective solution to compute the deformation field and deal with the remaining non-linear movements. Using a GPU the training only took

$\sim$ 10 minutes and once trained it could determine the deformation field in only a few seconds for hundreds of volumes whereas traditional registration would require multiple hours.

VI. References

References

[1] Guha Balakrishnan et al. “VoxelMorph: A Learning Framework for Deformable Medical Image Registration”. In: *IEEE Transactions on Medical Imaging* 38.8 (Aug. 2019), pp. 1788–1800. ISSN: 0278-0062, 1558-254X. DOI: 10.1109/TMI.2019.2897538. arXiv: 1809.05231. URL: http://arxiv.org/abs/1809.05231 (visited on 12/14/2020).

[2] M. Faisal Beg et al. “Computing Large Deformation Metric Mappings via Geodesic Flows of Diffeomorphisms”. In: *International Journal of Computer Vision* 61.2 (Feb. 1, 2005), pp. 139–157. ISSN: 1573-1405. DOI: 10.1023/B:VISI.0000043755.93987.aa. URL: https://doi.org/10.1023/B:VISI.0000043755.93987.aa (visited on 12/14/2020).

[3] B. B. Avants et al. “Symmetric diffeomorphic image registration with cross-correlation: Evaluating automated labeling of elderly and neurodegenerative brain”. In: *Medical Image Analysis*. Special Issue on The Third International Workshop on Biomedical Image Registration – WBIR 2006 12.1 (Feb. 1, 2008), pp. 26–41. ISSN: 1361-8415. DOI: 10.1016/j.media.2007.06.004. URL: http://www.sciencedirect.com/science/article/pii/S1361841507000606 (visited on 12/14/2020).

[4] Thomas Brox et al. “High Accuracy Optical Flow Estimation Based on a Theory for Warping”. In: *Computer Vision - ECCV 2004*. Ed. by Tomás Pajdla and Jiří Matas. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer, 2004, pp. 25–36. ISBN: 978-3-540-24673-2. DOI: 10.1007/978-3-540-24673-2_3.

[5] B. Jian and B. C. Vemuri. “Robust Point Set Registration Using Gaussian Mixture Models”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 33.8 (Aug. 2011). Conference Name: IEEE Transactions on Pattern Analysis and Machine Intelligence, pp. 1633–1645. ISSN: 1939-3539. DOI: 10.1109/TPAMI.2010.223.

[6] URL: https://colab.research.google.com/drive/1WiqyF7dCdnNBIANEY80Pxw_mVz4fyV-S?usp=sharing#scrollTo=Qbc-O66HPXNy (visited on 12/14/2020).