morphMan_docs Documentation

Release 0.1

Henrik Kjeldsberg

Oct 04, 2018

CONTENTS

Documentation for morphMan.

CHAPTER

ONE

INSTALLATION

morphMan is independent scripts which can be run from commandline, and not a python module. Installation is therefore simply to ensure that the dependencies are installed. Currently the project is only accessable through github.

1.1 Compatibility and Dependencies

The general dependencies of morphMan are

- VTK > 8.1
- Numpy > 1.13
- SciPy > 1.0.0
- VMTK 1.4

1.2 Basic Installation

Please confer with the homepage of each package for installation instructions. However, if you are on Linux or MaxOSX you can install all the packages through anaconda. First install Anaconda or Miniconda (preferably the python 3.6 version). Then execute the following command:

```
conda create -n your_environment -c vmtk python=3.6 itk vtk vmtk scipy numpy ipython
```

You can then activate your environment by runing `source activate your_environment`. You are now all set and can clone morphMan, and start manipulating your geometries:

```
git clone https://github.com/hkjeldsberg/vascularManipulationToolkit
```

CHAPTER

TWO

USING MORPHMAN

morphMan (morphological manipulation) is a collection of tools to objectivly manipulate morphological features of patient-specific vascular geometries. In the tutorials we examplify the usage by manipulating internal carotid arteries , but the tool can be applied to any geometry with a tubular shape.

The goal of morphMan is to provide researchers, and other users, with a set of tools to investigate the impact of altering morphological features in patient-specific geometries. For instance, morphMan can be used to investigate how cross-sectional area, bifurcation angles, ~~o~~ver all curvature change the local hemodynamics.

Although these features are possible to manually manipulate in advaced meshing software, morphMan is, to the best ouf our knowledge, the first *objective* and *reproduceble* method for varying a larger set of morphological features individually, while keeping the rest unchanged.

The basis for all our algorithms are centerlines and the Voronoi diagram of the surface. These 'representations' of the surface are easier to manipulate and controll than a surface where all cells are connected. An early version of a subset of the algorithms were presented in Bergersen[1] and Kjeldsberg[2].

The geometries used in the tutorials are taken from the Aneurisk repository[3], and are free for anyone to download. You can therefore easly follow the tutorials with the same geometries as we have used.

We would like to acknowledge the two open-source projects *VTK <https://www.vtk.org>* and *VMTK <http://www.vmtk.org>*, without this tool would not have been possible. Forthermore, we would also like to acknowledge the authors of Ford et al. 2011[4] for making the code from their publication open-source, and was the starting point for morphMan.

2.1 Geometric manipulation

The framework presented enable users to manipulate four geometric features independantly. Please see the tutorials for additional information:

- *Tutorial: Manipulate area* (create/remove a stenosis, in/decrease area variation, in/deflation of a vessel)
- *Tutorial: Manipulate bend* (Change the curvature or angle of a bend)
- *Tutorial: Manipulate bifurcation* (Change the angle in a bifurcation)
- *Tutorial: Manipulate curvature* (Increase or decrease the curvature variation in the a vessel segment)

[1] Bergersen, Aslak Wigdahl. Investigating the Link Between Patient-specific Morphology and Hemodynamics: Implications for Aneurism Initiation?. MS thesis. 2016.

[2] Kjeldsberg, Henrik Aasen. Investigating the Interaction Between Morphology of the Anterior Bend and Aneurysm Initiation. MS thesis. 2018.

[3] AneuriskWeb project website, http://ecm2.mathcs.emory.edu/aneuriskweb. Emory University, Department of Math&CS,i 2012.

[4] Ford, M.D., Hoi, Y., Piccinelli, M., Antiga, L. and Steinman, D.A., 2009. An objective approach to digital removal of saccular aneurysms: technique and applications. The British Journal of Radiology, 82(special_issue_1), pp.S55-S61.

2.2 New features

These four methods provide many degrees of freedom, however if you need a specific method or functionality, please do not hesitate to propose enhancements in the issue tracker, or create a pull request with new features.

CHAPTER THREE

TUTORIAL: MANIPULATE AREA

Manipulation of the cross-sectional area is performed by running the script `manipulate_area.py`. In this tutorial we are using the model with ID C0001 from the Aneurisk database. For the commands below we assume that there is a folder *./C0001* with the file *surface.vtp*, relative to where you execute the command.

For changing the area you first need to define which segment you want to alter. For `manipulating_area.py` there are three choises:

- Manuall selection, based on clicking on a surface
- Provide the points on the centerline
- Pass the argument `--region-of-interest first_line`, which simply choses the section between the inlet

and the first bifurcation. Note: This options is not possible for creating or removing a stenosis.

There are three different methods for altering the cross-sectional area:

- Increase or deacrease the area variation
- Inflation of deflation of a vessel
- Create or remove a stenosis

`manipulate_area.py` is easy to extend with new features. Please make a pull request with changes, or contact the developers for help with extending the method.

3.1 Area variations

The goal of this method is to increase or decrease the variation of the cross-sectional area along an arterial segment. Shown in Figure 1 drawing of the desired output. More speceficly, we would like to control R, the ration between the largest and smallest area. The area is changed with a factor F, which is controlled by β.

For changing the cross-sectional area variation you can either provide the desired R you want (`--ratio`), or specify β (`--beta`). Note that $\beta > 0$ increases, and $\beta < 0$ decreases the variability.

In order to output a plausible geometry the the first and last 10 % is a linear transition between the manipulated and original geometry. However, if you choose `--region-of-interest first_line`, only the end of the line changes.

Figure 2, you can see the output of the algorithm, where the internal coratid artery was is defined as the region of interest, and were obtain by running:

```
python area_variations.py --ifile C0001/surface.vtp --ofile C0001/increased_variation.
↪vtp --method variation --ratio 3.0 --region_of_interest first_line
```

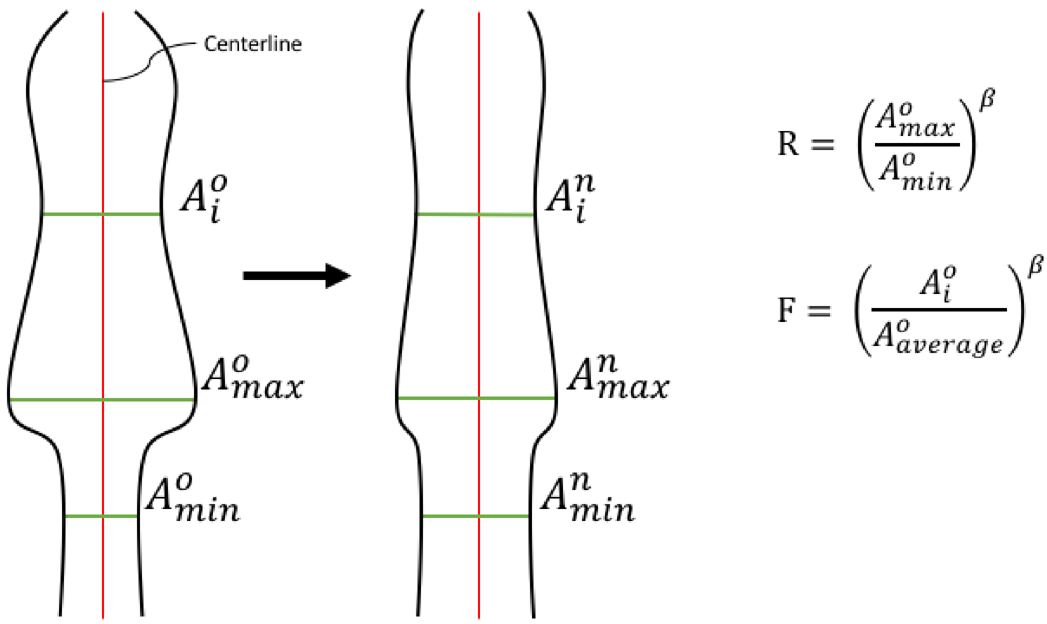


Fig. 1: Figure 1: A drawing of what we would like to happen.

For the model with increased cross-sectional area variation, and:

```
python area_variations.py --ifile C0001/surface.vtp --ofile C0001/increased_variation.
↪vtp --method variation --ratio 1.4 --region_of_interest first_line
```

For the model with decreased cross-sectional area variation.

3.2 Create / remove a stenosis

A stenosis is a local narrowing of the vessel, often caused by the buildup of plauqe. You can manipulate a stenosis with `manipulate_area.py` by setting `--method stenosis`.

3.2.1 Create a stenosis

For creating a stenosis you need to define the center, either by providing the point manually, or using `--stenosis-points`. The stenosis will be upstream and downstream of the center, with `--length` times the local minimal inscribed sphere, and `--percent` controlls how much the local radius chould be changed. For instance, if `--percent 50` is provided as an argument, the stenosis will have an area in the midle of stenosis of $(0.5r)^2\pi$, where r is the radius of the area in the original geometry.

The stenosis is now assumbed to have sinusodial shape. However, altering the shape is trivial, and can be done by adapting one line in the function `get_factor` in `manipulate_area.py`.

In Figure 3 you can see the output of the script. Of note is that creating a stenosis in sections with high curvature, like the carotid siphon, is unproblematic.

To recreate the above output, execute the following on the commandline line:

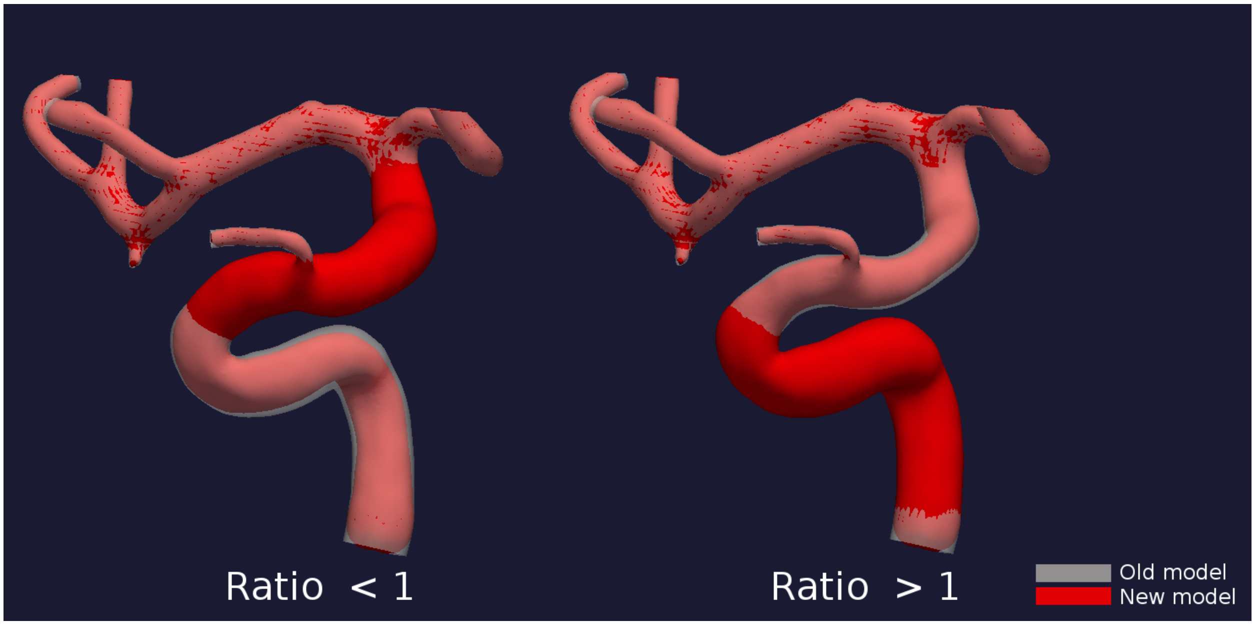


Fig. 2: Figure 2: Area variations throughout the geometry for different ratios.

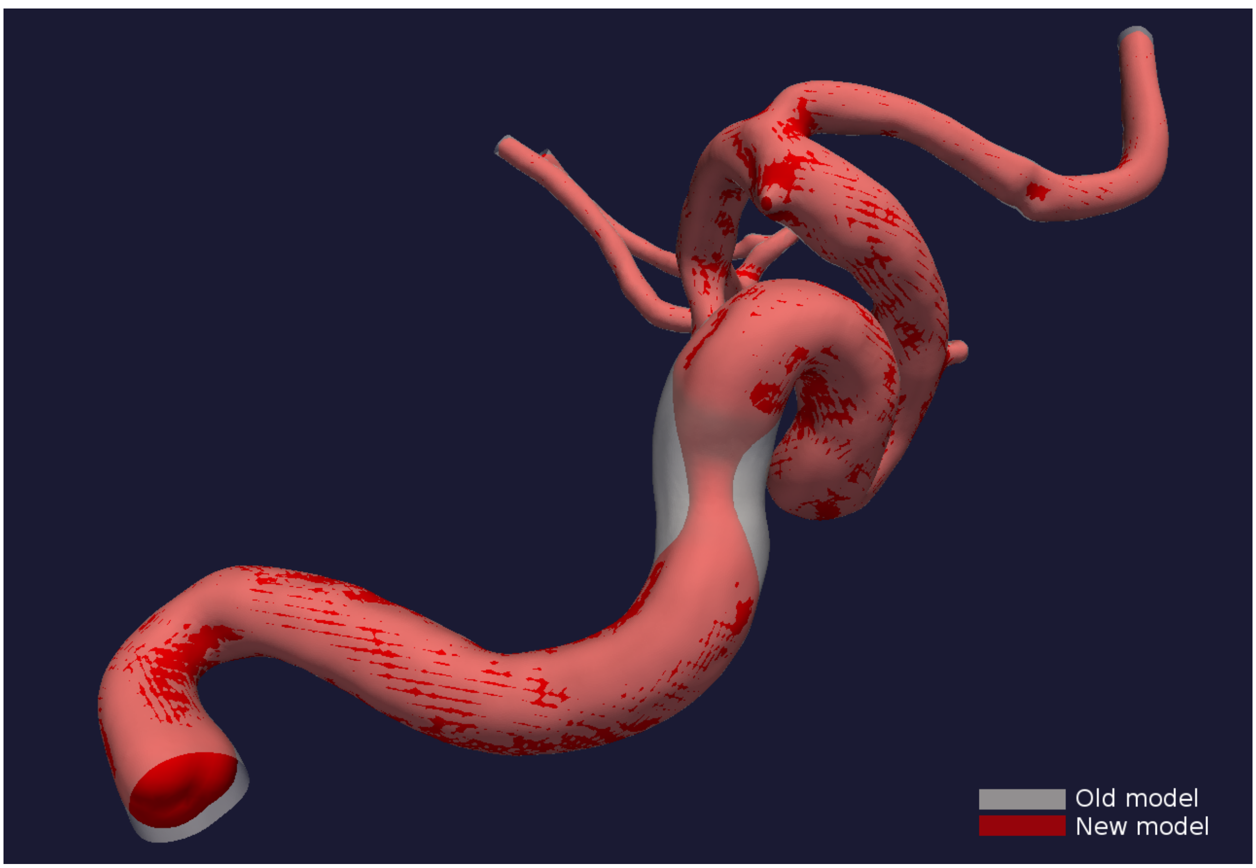


Fig. 3: Figure : Comparison of new and old model, with and without stenosis.

```
python manipulate_area.py --ifile C0001/surface.vtp --ofile C0001/stenosis.vtp --
↪smooth True --method stenosis --stenosis-point x y z --percentage 50 --size 1
```

3.2.2 Remove stenosis

To remove a stenosis, you need to provide two points, one at each end of the stenosis. The area will be changed to alter linearly over the chosen region of interest.

To exemplify this, we can use the output from the previous example `C0001/stenosis.vtp`. To the left in Figure 4a you can see the stenosed model compared to the original, and to the right the comarison of the original model and the one were we have removed the stenosis.

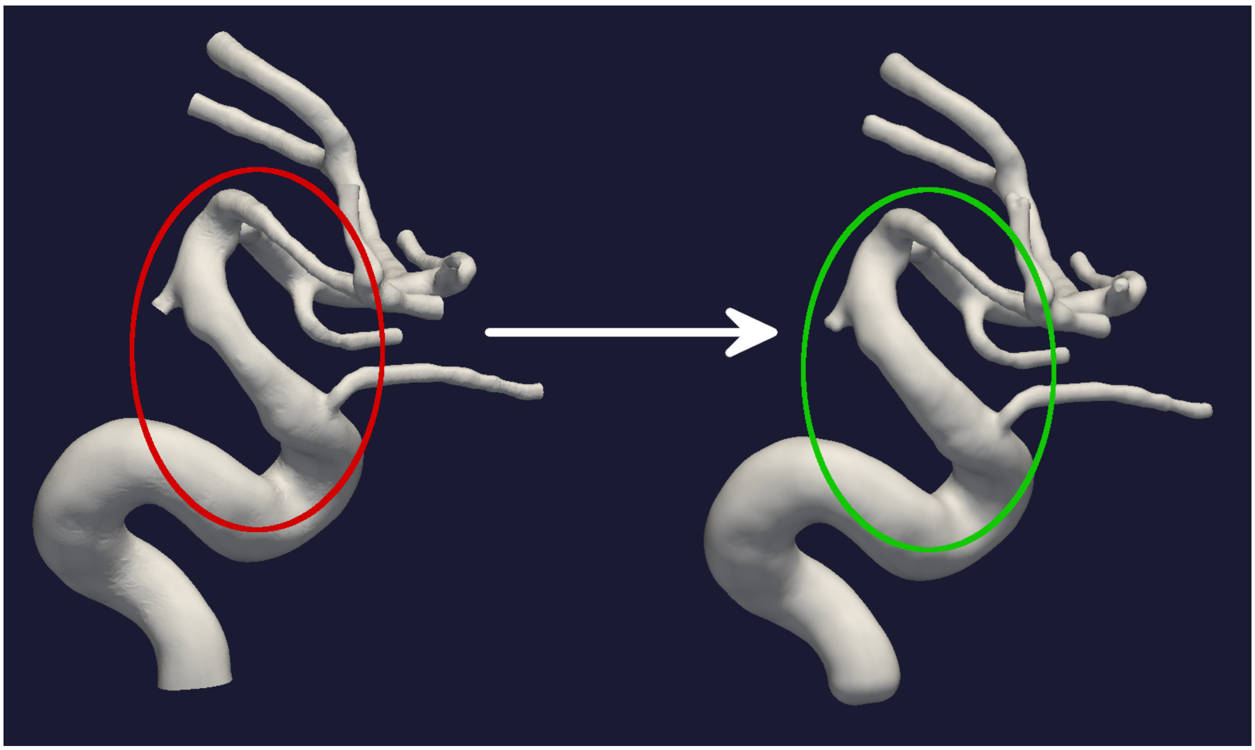


Fig. 4: Figure : Comparison of new and old model, with and without stenosis.

To reproduce the above result, execute the following command:

```
python manipulate_area.py --ifile C0001/stenosis.vtp --ofile C0001/no_stenosis.vtp --
↪smooth True --method stenosis --stenosis-point x y z x y z
```

3.3 Inflation / deflation of an arterial segment

The area of interest can also be inflated or decflated. To do so, pass the argument `--method area`, and set the percentage change with `--percentage`. Like with *Area variations* the first and last 10 % of the region of interest is a transition between the original and modified geometry to ensure smooth transitions.

To perform a deflation run the following command:

```
python manipulate_area.py --ifile C0001/surface.vtp --ofile C0001/inflated.vtp --
↪smooth True --percentage -20 --method area --region-of-interest first_line
```

~~Below~~ is an illustration of area decrease and increase in a single patient-specific model.

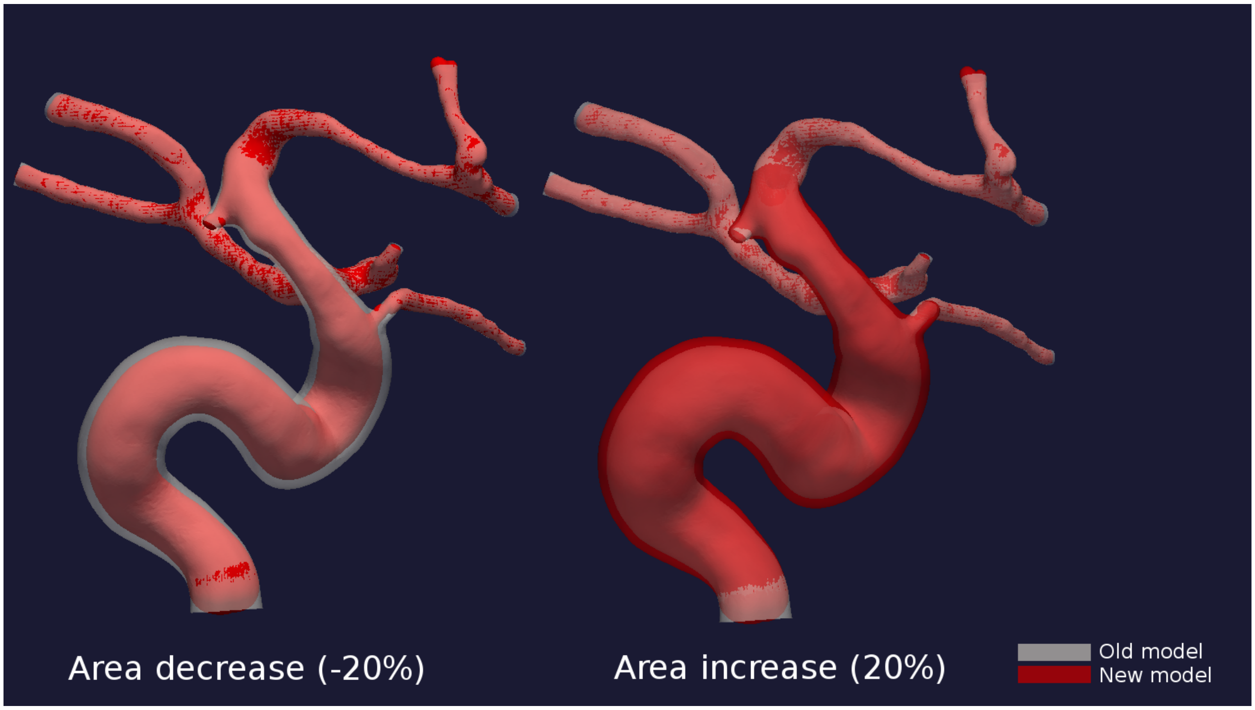


Fig. 5: Figure : Decrease and increase in overall area.

For additional information, beyond this tutorial, on the script and input parameters please run `python manipulate_area.py -h` or confere with the *API documentation*.

CHAPTER

FOUR

TUTORIAL: MANIPULATE BEND

Note:

Can be used for a general bend, but if used in ICA... Manipulation is initialized by selecting a segment of the vessel, bounded by two clipping points.

The two clipping points can be freely chosen along the centerline, but it is highly recommended to landmark the geometry in order to objectively segment the geometry, and use the resulting landmarking points as clipping points.

Adjusting curvature and angle in the anterior bend utilizes a common script: `move_siphon.py`. The script performs geometric manipulation of the anterior bend segment, as defined in the landmarking section. Adjusting the anterior bend relies only on two parameters, the compression/extension factors α and β. Alteration of the curvature or angle of the anterior bend is performed by specifying these factors in the script `automated_geometric_quantities.py` and `calculate_alpha_beta_values.py`. The pipeline for increasing or decreasing either curvature or the bend angle in the anterior bend is described below.

Alternatively the user may choose any arbitrary values for α and β.

To perform geometric manipulation of the anterior bend, run the following command:

```
python move_siphon.py --dir_path [PATH_TO_CASES] --case [CASENAME] --alpha [ALPHA] --
↪beta [BETA]
```

In general, the compression / extension factors α and β determine the magnitude and direction in which the anterior bend is translated. The manipulation script allows movement in two directions:

- Vertical, determined by α
- Horizontal, determined by β

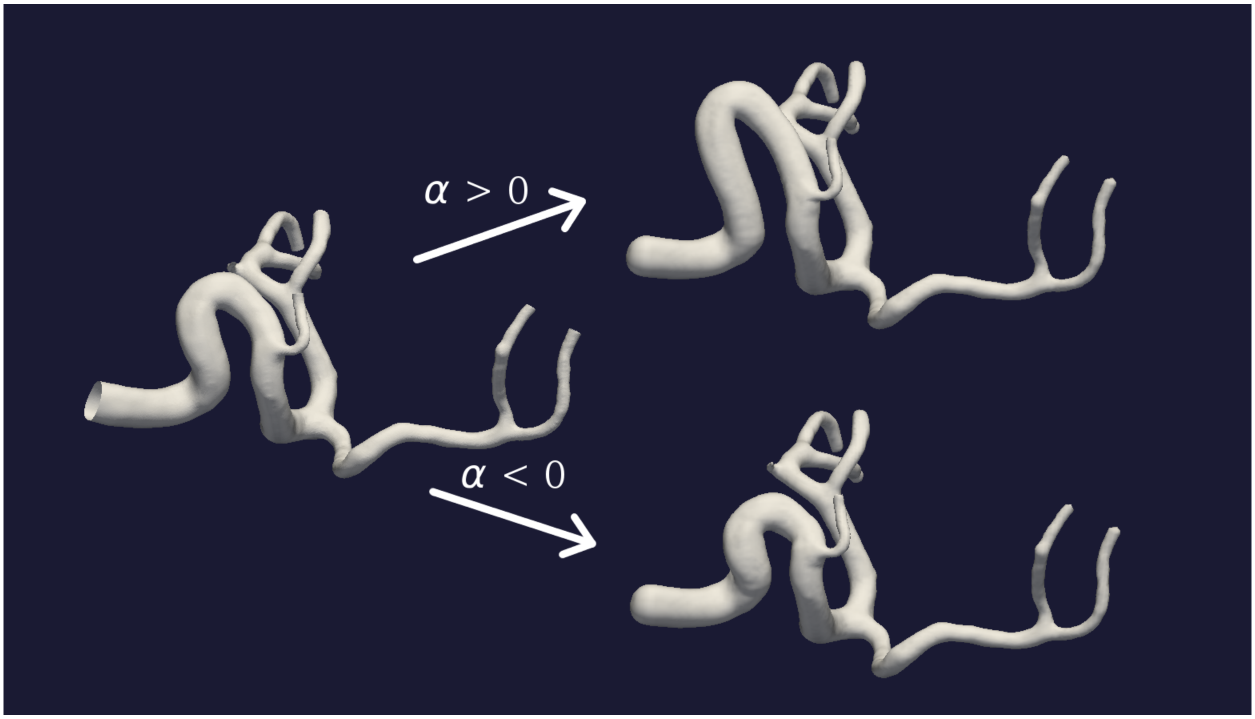


Fig. 1: Figure 6: Movement in the vertical direction, determined by α. FIXME: New model, old model.

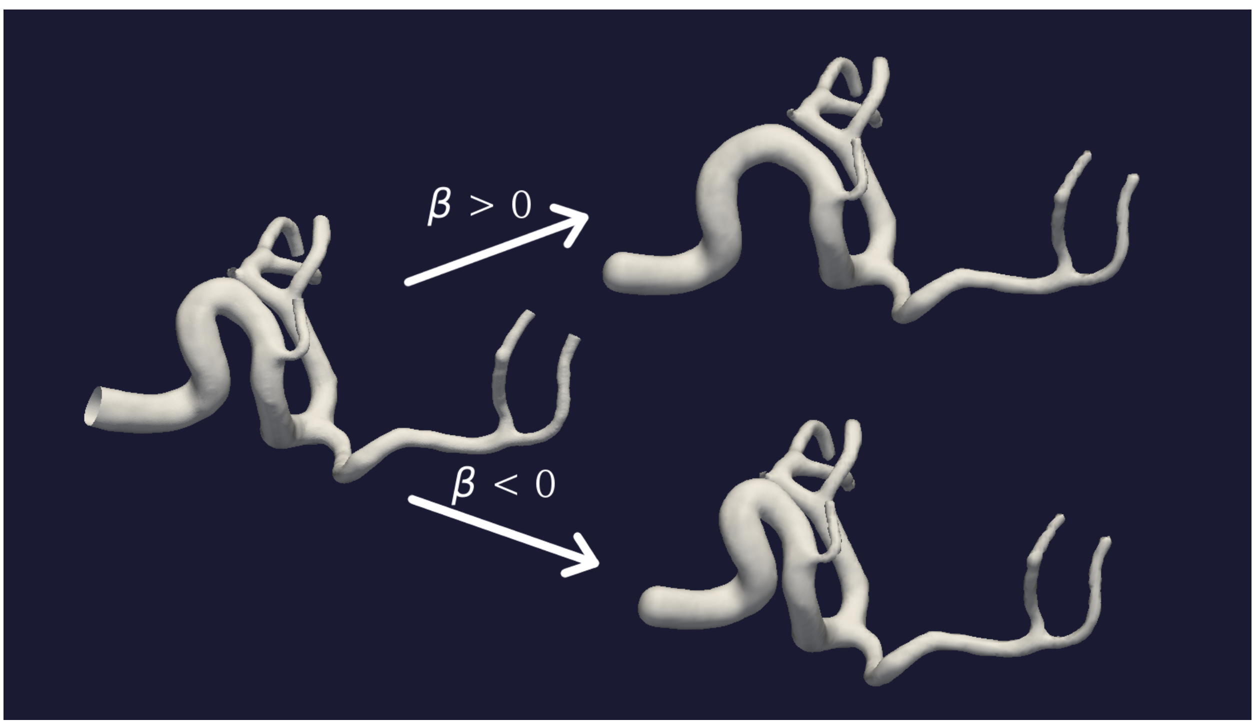
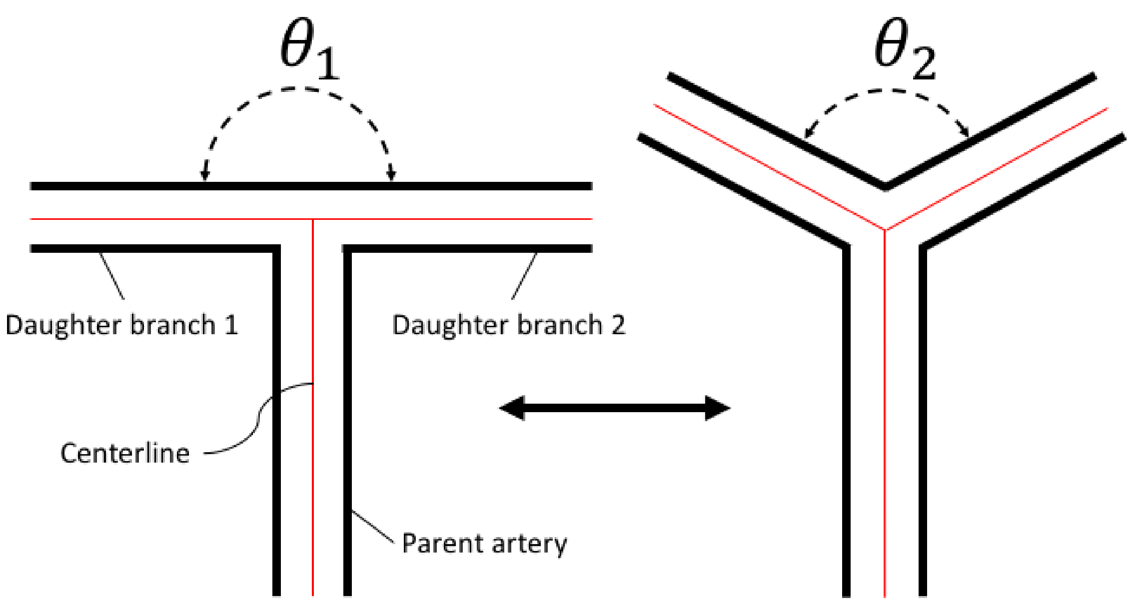


Fig. 2: Figure 7: Movement in the horizontal direction, determined by β. FIXME: New model, old model.

CHAPTER FIVE

TUTORIAL: MANIPULATE BIFURCATION

The goal with `manipulate_bifurcation.py` is to control the angle between two daughter branches in a bifurcation, see Figure 1. The daughter branches can be rotated towards each other, in other words reduce θ as defined in Figure 1, or towards the parent artery, increasing θ. The algorithm builds on previous work of Ford et al.[1]



~~Fig. 1:~~ Figure 1: An illustration of what manipulate_bifurcation.py does.

In this tutorial we are using the model with ID C0003 from the Aneurisk database. For the commands below we assume that there is a folder *./C0003* with the file *surface.vtp*, relative to where you execute the command.

The default is to rotate both branches, but if either `--keep-fixed1` or `--keep-fixed2` is set to **True**, daughter branch 1 or 2 will be keept fixed, respectivly. Furthermore, if both parameters are set to **True** then the algorithm can be used to remove an aneurysm (a balloon-shaped bleb on the artery).

Shown in Figure 2 is the result of two rotating the daughter branched with both a positive and negative angle.

To reporduce the results, please run the following commands:

[1] Ford, M.D., Hoi, Y., Piccinelli, M., Antiga, L. and Steinman, D.A., 2009. An objective approach to digital removal of saccular aneurysms: technique and applications. The British Journal of Radiology, 82(special_issue_1), pp.S55-S61.

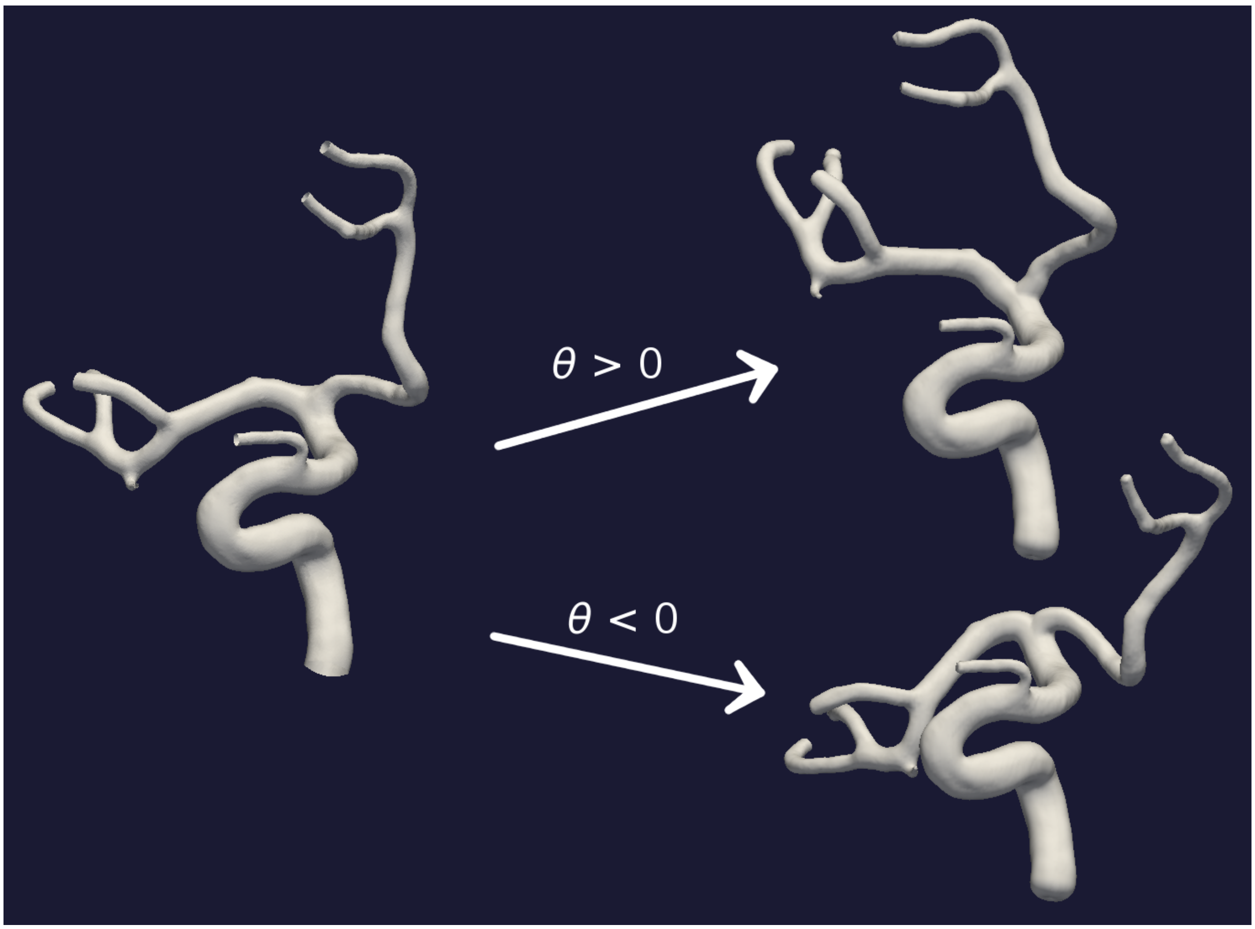


Fig. 2: Figure 2: Rotation of daughter branches, in both a widening and narrowing of the bifurcation angle.

```
python manipulate_bifurcation.py --ifile C0003/surface.vtp --ofile C0003/rotate_pluss.vtp --angle -20
python manipulate_bifurcation.py --ifile C0003/surface.vtp --ofile C0003/rotate_minus.vtp --angle -20
```

Inspecting Figure 2 closely you can observe an unphysiological "notch" in the bifurcation of the surface with increased θ. On remedy is to add the flag `--bif True` and `--lower True`, which will output a smoother bifurcation, see Figure 3. Using both flags haven proven to give an improved surface, and when used for computational fluid dynamics, a more physiological wall shear stress[2].

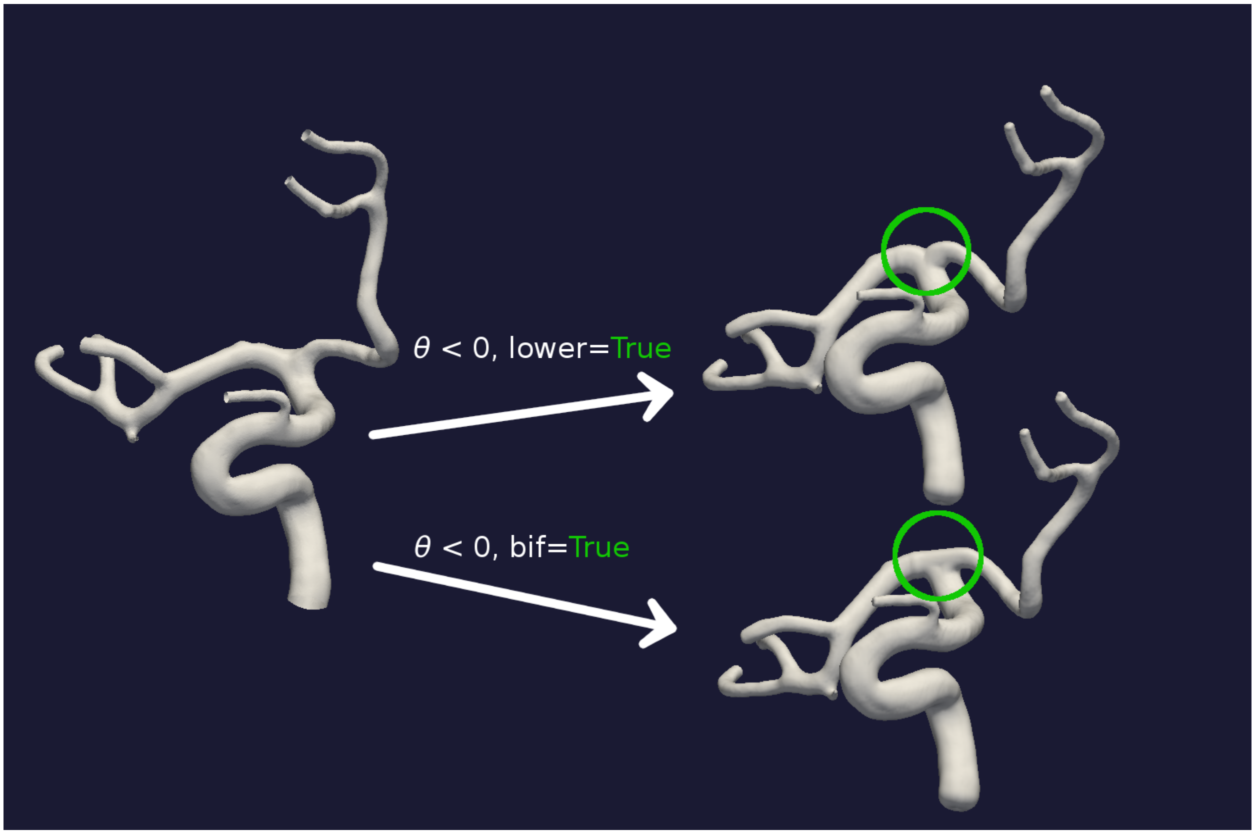


Fig. 3: Figure 3: Rotation of daughter branches with different reconstruction of the bifurcation.

For additional information, beyond this tutorial, on the script and input parameters please run `python manipulate_bifurcation.py -h` or confere with the *API documentation*.

[2] Bergersen, A.W., Chnafa, C., Piccinelli, M., Gallo, C., Steinman, D.A., and Valen-Sendstad, K. In preparation. Automated and Objective Removal of Bifurcation Aneurysms: Incremental Improvements?

CHAPTER

SIX

TUTORIAL: MANIPULATE CURVATURE

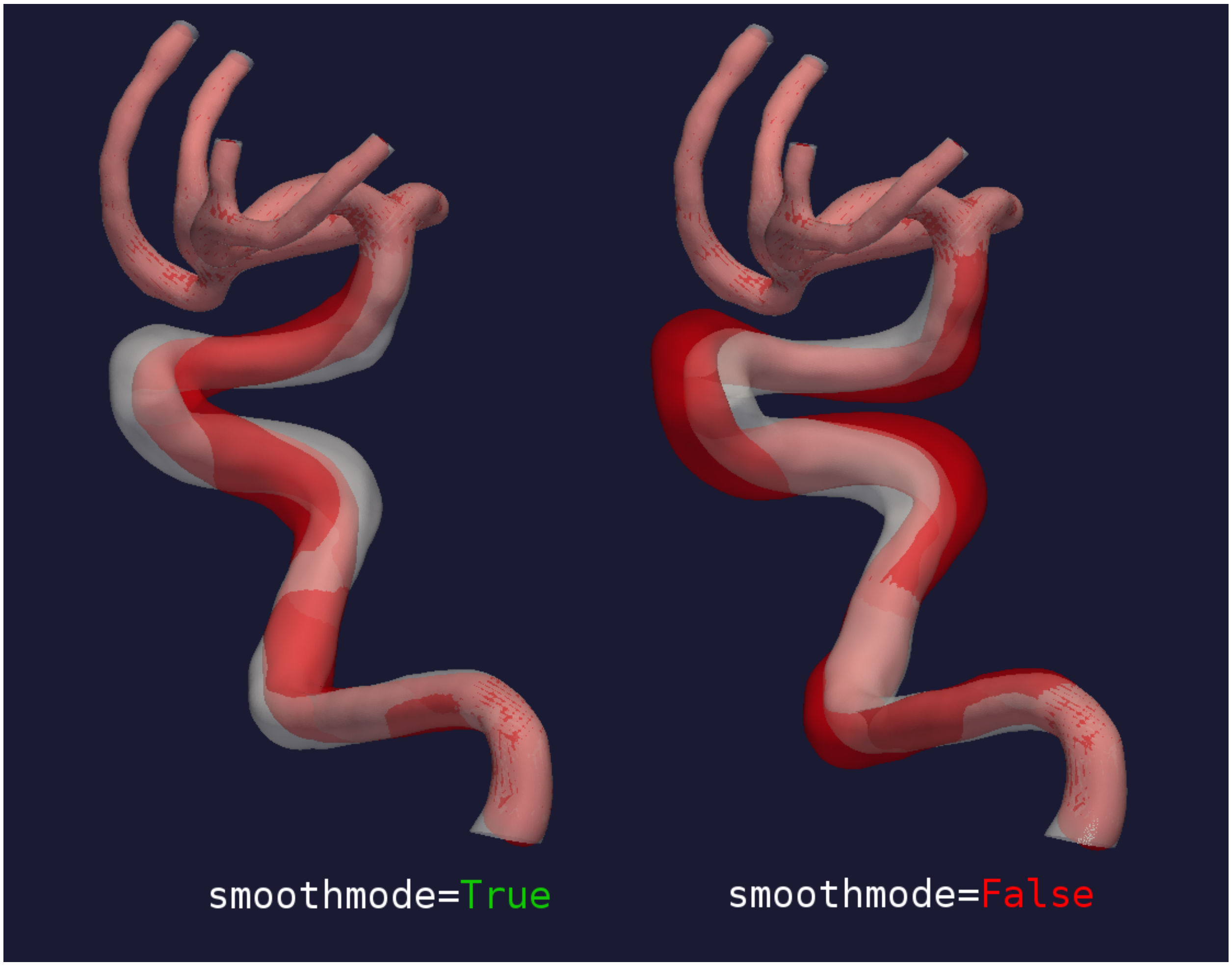


Fig. 1: Figure 7: Sharpened and smoothened version of the siphon.

CHAPTER

SEVEN

MISCELLANEOUS

7.1 Landmarking of the carotid siphon

Landmarking of the geometry is performed in order to identify different segments of the vessel. Identification of specific segments is required in order to perform geometric manipulation. The script `automated_landmarking.py` includes implementation of two automated landmarking algorithms; the first introduced by Piccinelli et al. (2011), the second by Bogunović et al. (2014).

Both landmarking algorithms rely on geometric properties of the centerline. The script allows the user to select one of four different methods to compute the curvature of the discrete centerline:

1. B-Splines (`spine`)
2. Free-knot regression splines (`freeknot`)
3. Discrete derivatives (`disc`)
4. VMTK (`vmtk`)

Todo: Add references to different splining techniques?

To perform landmarking, run the following command:

```
python automated_landmarking.py --dir_path [PATH_TO_CASES] --algorithm [ALGORITHM] --
↪curv_method [METHOD]
```

This will produce `landmark_ALGORITHM_CURVMETHOD.particles` which contains four points defining the interfaces between the segments of the vessel.

7.2 Selection of compression / extension factors

The compression / extension factors α and β determine the magnitude and direction in which the anterior bend is translated. Running the scripts `automated_geometric_quantities.py` and `calculate_alpha_beta_values.py` is required if the user is interested in reaching a spesific change in angle or curvature.

7.3 Common

Wrapping vtk and vmtk functionalty.

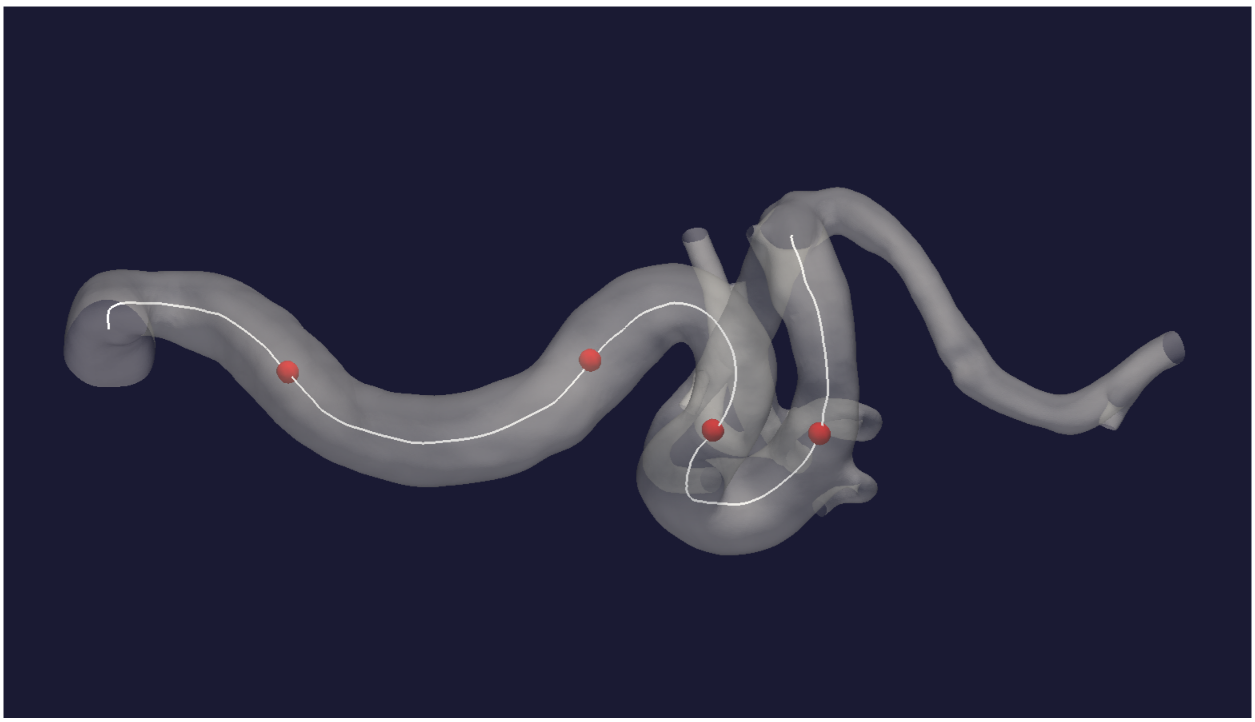


Fig. 1: Figure 4: Landmarked geometry, with interfaces shown as red spheres.

CHAPTER

EIGHT

API DOCUMENTATION

8.1 Manipulation of a bend

8.1.1 manipulate_bend.py

8.2 Manipulation of a the area

8.2.1 manipulate_area.py

8.3 Manipulation of a curvature

8.3.1 manipulate_curvature.py

8.4 Manipulation of a bifurcation

8.4.1 manipulate_bifurcation.py

8.5 Miscellaneous

Supporting functions to the main algorithms.

8.5.1 common.py

8.5.2 automated_landmarking.py

8.5.3 automated_geometric_quantities.py

8.5.4 calculate_alpha_beta_values.py