# VICTRE Breast Phantom Documentation

***Release 1.0***

**Christian G. Graff**

**Mar 28, 2019**

# Contents:

Welcome to the VICTRE Breast Phantom documentation!

This documentation provides information regarding how to download, install, and use the VICTRE Breast Phantom which is designed to generate random anthropomorphic voxelized whole-breast phantoms.

This software was developed as part of the Virtual Imaging Clinical Trials for Regulatory Evaluation (VICTRE) project at the U. S. Food and Drug Administration.

The VICTRE tools provide a complete simulated imaging chain for mammography and digital breast tomosynthesis and are available as open source software

CHAPTER 1

# Installation

The VICTRE Breast Phantom source code is available here. It is written in C++ and therefore must be compiled for your system before use.

## 1.1 Operating System

This software has been developed and tested on Debian and Red Hat based Linux distributions. Compilation is via cmake and should be possible on any UNIX-like operationg system (including MacOS) with minimal modification to the build configuration in the provided CMakeLists.txt file, however the following distributions are recommended:

- Ubuntu 16.04
- Centos 7.x

The following build instructions are based on Ubuntu 16.04.

## 1.2 Compilation

First, install the packages necessary to compile the software (reminder: instructions are specific to Ubuntu, consult your operating system documentation for analagous package names and installation instructions) From a terminal command prompt:

```
> sudo apt-get install cmake vtk6 libvtk6-dev \
                     libblas-dev liblapack-dev \
                     libboost-dev libboost-program-options-dev \
                     libproj-dev zlib1g-dev
```

Next, clone the git repository:

```
> git clone https://github.com/DIDSR/breastPhantom
```

From a suitable build directory generate a Makefile using cmake and run make:

```
> cmake <path to repository>
> make
```

where <path to repository> is the directory containing the cloned repository.

**Note:** when running cmake you may recieve an error message related to references to files "/usr/lib/x86_64-linux-gnu/libvtkRenderingPythonTkWidgets.so" and "/usr/bin/vtk". This is a known VTK bug but since we are not linking to these files, the error can safely be ignored.

## 1.3 Installation

If the previous steps were successful, the executable breastPhantom should be present in the build directory. This file can be copied to a convenient location. No additional installation steps are required. Consult the next section which describes how to run the executable.

# CHAPTER 2

# Running the Software

The executable breastPhantom is run from the command line and takes a single required command line argument:

```
> breastPhantom -c [FILE]
```

where [FILE] is a configuration file which specifies all the breast model parameters.

## 2.1 Configuration file

Several sample configuration files are provided, all ending with the extension .cfg. Configuration files can be modified with your favorite text editor. A configuration file has many parameters, please see <ref> for a complete list with brief explanations.

A few of the parameters that most users will probably want to modify are the following:

***outputDir*** The directory where the generated phantom files will be written

***imgRes*** The phantom voxel size (in mm). Typically this varies between 0.05 and 0.25 depending on application

***targetFatFrac*** The desired breast density in terms of volume fractional fat content. Should be between 0 and 1, though for typical breasts should vary between approximately 0.4 and 0.95.

All configuration parameters are explained <here>

## 2.2 Generating a phantom

Once the desired configuration file is created, the software is run from the command line as above. Several status messages will be generated during the phantom creation process. The time required to generate a phantom can vary significantly and is highly dependent on the voxel size (imgRes parameter) and the processing capabilities of your computer. A low resolution phantom on a fast computer may require 30 mins of processing time, while a high-resolution phantom on a slow computer may take more than a day. Sections of this software are multi-threaded and should take advantage of multiple cores if available. The number of cores used can be controlled with the environment variable OMP_NUM_THREADS if desired. All cores will be used by default.

All phantom output files will be placed in the directory specified in the configuration file

## 2.3 Randomization

The phantom generation process is stochastic and relies upon a random number generator seed. This seed can be specified in the configuration file using the parameter *seed*, though the default behavior is to randomly generate a seed at runtime by reading from */dev/urandom*. In this mode each time the breast phantom generation code is run with a fixed configuration file, a different random breast will be generated that is consistent with the configuration parameters. This allows for the generation of random breast textures at fixed glandularity, for example.

# CHAPTER 3

# Output Formats

## 3.1 Voxel Values

The main output of the software is a 3D voxelized breast phantom where each voxel is an unsigned 8 bit integer tissue label (possible values 0-255). Here is a lookup table specifying the labels for each tissue type.

| Tissue | Label |
|---|---|
| air | 0 |
| fat | 1 |
| skin | 2 |
| glandular | 29 |
| nipple | 33 |
| muscle | 40 |
| ligament | 88 |
| TDLU[1] | 95 |
| duct | 125 |
| artery | 150 |
| vein | 225 |
| cancerous mass[2] | 200 |
| calcification[2] | 250 |
| compression paddle[2] | 50 |

[1]Terminal Duct Lobular Unit

[2]These tissues are not created by the phantom generation software, but can be added by other VICTRE pipeline software. They are included here for completeness.

## 3.2 Output Files

For each run of the breast phantom generation software several files are created in the output directory specified in the configuration file. The naming convention is that all output filenames begin with `p_` followed by the random number generator seed that was used and an extention specifying the type of output file. Here is a list of output files:

**p_*nnnnnnnn*.cfg** A copy of the configuration file used to generate the phantom

**p_*nnnnnnnn*.vti** The phantom volume stored in VTK image format (XML with zlib-compressed data element). This is a file format native to the Visualization Toolkit. See https://www.vtk.org for more information. This file can be opened in Paraview, an open-source, multi-platform data analysis and visualization application available for download at https://www.paraview.org/

**p_*nnnnnnnn*.raw.gz** The raw phantom volume stored as 8-bit unsigned integers in a gzip archive. There is no file header.

**p_*nnnnnnnn*.mhd** A metaImage header file containing information about the phantom data stored in the file p_*nnnnnnnn*.raw.gz Parsing this header file will allow you to read and manipulate raw.gz phantom files. See https://itk.org/Wiki/MetaIO/Documentation for more information on the MetaImage format

**p_*nnnnnnnn*.loc** This file lists candidate locations for insertion of masses and calcifications. Each location is where a TDLU has been randomly generated. Terminal duct lobular units are a common site for cancer formation. Locations are stored in plain text, one location per line with comma separated coordinates in millimeters. See the phantom geometry section for an explanation of the coordinate system.

# CHAPTER 4

## Geometry and Visualization

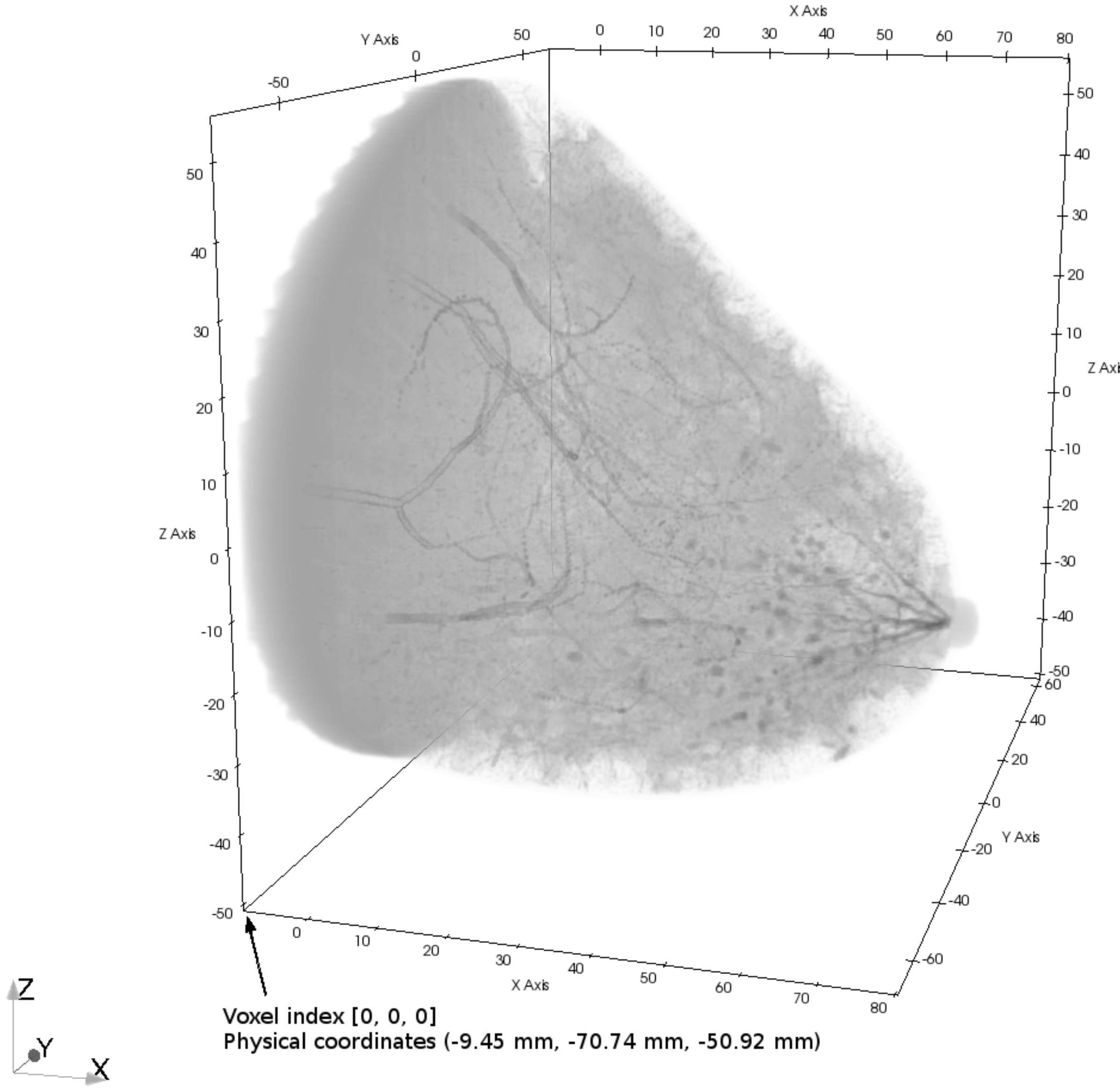
### 4.1 Visualization Software

A variety of software is available for visualizing phantom data. Raw data can be read using any programming language, Python, Matlab, etc. Here are some tips.

- Paraview https://www.paraview.org can read .vti files and .mhd headers. To read the phantom using .mhd header the .raw.gz file must first be extracted with gunzip
- When viewing phantoms in Paraview you may run into memory limitations. Uncompressed high resolution phantoms may exceed 8GB and cause Paraview to run slowly or crash. Slice view is more reliable than volume view for visualizing large phantoms in Paraview.
- Other software that may be useful for visualizing phantom files: ImageJ https://imagej.nih.gov/ij/ , Amide http://amide.sourceforge.net/
- Large phantoms may cause memory problems and should be loading into memory directly as 8 bit integers when possible.
- Large phantoms may contain more than $2^{32}$ voxels, causing problems in for loops. Use of 64 bit integer indexing variables may be necessary.

### 4.2 Coordinate System

Each phantom has a physical coordinate system (millimeter units) and a voxel indexing scheme. The following figure illustrates the connection between these two systems.




- The *x* coordinate (first voxel index) is the chest wall to nipple direction
- The *y* coordinate (second voxel index) is the lateral (left-right) direction
- The *z* coordinate (third voxel index) is the inferior-superior direction
- Voxel index [0, 0, 0] is noted in the figure in the most inferior transverse plane

The corresponding mhd header file for this phantom contains the following lines:

```
Offset = -9.4500 -70.7366 -50.9244
ElementSpacing = 0.1000 0.1000 0.1000
DimSize = 899 1351 1068
```

This indicates the physical location of the voxel with index [0, 0, 0] is at physical location (-9.4500mm, -70.7366mm, -50.9244mm). The total number of voxels in the x, y, and z dimensions are 899, 1351, and 1068 respectively. Each voxel is a 0.1 mm cube.

# CHAPTER 5

## Sample Configuration Files

The VICTRE project made use of four configuration files to represent four breast density levels. These configuration files can be found in the cfg directory of the github repository. In the VICTRE study 50 micron voxel size was used. For testing purposes it is advised to use the 100 micron voxel size configuration files as the factor of 8 reduction in total number of voxels reduces the phantom generation time by a similar factor. For further details of the VICTRE trial patient population please consult the associated paper and suplementary material

**Note:** These four configuration files do not capture the true diversity in breast anatomy and are meant to be starting points for further investigations of appropriate variations in configuration parameters

CHAPTER 6

# Configuration Files Parameters

The configuration file contains all the parameters necessary to generate a breast phantom. They are divided into several categories, primarily by tissue type. In the following tables each parameter is given a brief description.

## 6.1 base parameters

| Name | Type | Notes |
|---|---|---|
| base.outputDir | string | directory output files are saved to |
| base.imgRes | float (mm) | voxel size |
| base.skinThick | float (mm) | thickness of skin |
| base.nippleLen | float (mm) | length of nipple |
| base.nippleRad | float (mm) | radius of nipple |
| base.areolaRad | float (mm) | radius of thicker skin around nipple |
| base.leftBreast | Boolean | true for left breast, false for right breast |
| base.targetFatFrac | float (mm) | desired fraction of interior breast volume containing fat |
| base.seed | integer | random number seed (chosen randomly if not specified) |

## 6.2 shape parameters

| Name | Type | Notes |
|---|---|---|
| shape.ures | float | mesh resolution for breast shape |
| shape.vres | float | mesh resolution for breast shape |
| shape.pointSep | float (mm) | minimum point separation for point cloud |
| shape.ringWidth | float (mm) | thickness of muscle backing layer |
| shape.ringSep | float (mm) | mesh resolution for muscle layer |
| shape.featureAngle | float (degrees) | minimum angle to preserve during mesh smoothing |
| shape.targetReduction | float | fraction of triangles to remain after decimation |

Continued on next page

Table 1 – continued from previous page

| Name | Type | Notes |
|---|---|---|
| shape.a1b | float | scale of breast bottom |
| shape.a1t | float | scale of breast top |
| shape.a2l | float | scale of breast left side |
| shape.a2r | float | scale of breast right side |
| shape.a3 | float | breast outward scale |
| shape.eps1 | float | u direction quadric shape exponent |
| shape.eps2 | float | v direction quadric shape exponent |
| shape.doPtosis | Boolean | if true include ptosis in shape |
| shape.ptosisB0 | float | ptosis parameter B0 |
| shape.ptosisB1 | float | ptosis parameter B1 |
| shape.doTurn | Boolean | if true include turn deformation |
| shape.turnC0 | float | turn deformation parameter C0 |
| shape.turnC1 | float | turn deformation parameter C1 |
| shape.doTopShape | Boolean | if true include top shape deformation |
| shape.topShapeS0 | float | top shape deformation parameter S0 |
| shape.topShapeS1 | float | top shape deformation parameter S1 |
| shape.topShapeT0 | float | top shape deformation parameter T0 |
| shape.topShapeT1 | float | top shape deformation parameter T1 |
| shape.doFlattenSide | Boolean | if true include flatten side deformation |
| shape.flattenSideG0 | float | flatten side deformation parameter G0 |
| shape.flattenSideG1 | float | flatten side deformation parameter G1 |
| shape.doTurnTop | Boolean | if true include turn top deformation |
| shape.turnTopH0 | float | turn top deformation parameter H0 |
| shape.turnTopH1 | float | turn top deformation parameter H1 |

## 6.3 glandular compartment parameters

| Name | Type | Notes |
|---|---|---|
| compartments.num | integer | number of glandular compartments |
| compartments.seedBaseDist | float (mm) | distance along nipple line of compartment seed base |
| compartments.backFatBufferFrac | float | fraction of breast at muscle layer forced to be fat |
| compartments.numBackSeeds | integer | number of back plane Voronoi seeds |
| compartments.angularJitter | float | compartment Voronoi seed jitter (fraction of subtended angle) |
| compartments.zJitter | float (mm) | maximum compartment Voronoi seed jitter in direction of nipple |
| compartments.maxFracRadialDist | float | maximum radial distance from base seed as a fraction of distance to breast surface |
| compartments.minFracRadialDist | float | minimum radial distance from base seed as a fraction of distance to breast surface |
| compartments.minScaleNippleDir | float | minimum scale in nipple direction |
| compartments.maxScaleNippleDir | float | maximum scale in nipple direction |
| compartments.minScale | float | minimum scale in non-nipple direction |
| compartments.maxScale | float | maximum scale in non-nipple direction |
| compartments.minGlandStrength | float | minimum gland strength |
| compartments.maxGlandStrength | float | maximum gland strength |
| compartments.maxDeflect | float | maximum compartment deflection angle from pointing towards nipple (fraction of pi) |
| compartments.minSkinScaleNippleDir | float | minimum scale skin seeds in nipple direction |
| compartments.maxSkinScaleNippleDir | float | maximum scale skin seeds in nipple direction |
| compartments.minSkinScale | float | minimum scale skin in non-nipple direction |
| compartments.maxSkinScale | float | maximum scale skin in non-nipple direction |
| compartments.skinStrength | float | skin strength |
| compartments.backScale | float | back scale |
| compartments.backStrength | float | back strength |
| compartments.nippleScale | float | nipple scale |
| compartments.nippleStrength | float | nipple strength |
| compartments.voronSeedRadius | float (mm) | radius from point to check Voronoi seed distance |

## 6.4 TDLU parameters

| Name | Type | Notes |
|---|---|---|
| TDLU.maxLength | float (mm) | maximum TDLU length |
| TDLU.minLength | float (mm) | minimum TDLU length |
| TDLU.maxWidth | float (mm) | maximum TDLU width |
| TDLU.minWidth | float (mm) | minimum TDLU width |

## 6.5 Perlin noise parameters

| Name | Type | Notes |
|---|---|---|
| perlin.maxDeviation | float | maximum fraction of radius deviation |
| perlin.frequency | float | starting frequency |
| perlin.lacunarity | float | octave frequency multiplier |
| perlin.persistence | float | octave signal decay |
| perlin.numOctaves | integer | number of frequency octaves |
| perlin.xNoiseGen | integer | x direction noise generation seed |
| perlin.yNoiseGen | integer | y direction noise generation seed |
| perlin.zNoiseGen | integer | z direction noise generation seed |
| perlin.seedNoiseGen | integer | seed noise generation |
| perlin.shiftNoiseGen | integer | shift noise generation seed |

## 6.6 Compartment boundary noise parameters

| Name | Type | Notes |
|---|---|---|
| boundary.maxDeviation | float | maximum fraction of distance deviation |
| boundary.frequency | float | starting frequency |
| boundary.lacunarity | float | octave frequency multiplier |
| boundary.persistence | float | octave signal decay |

## 6.7 fat lobule boundary perturbation noise parameters

| Name | Type | Notes |
|---|---|---|
| perturb.maxDeviation | float | maximum fraction of distance deviation |
| perturb.frequency | float | starting frequency |
| perturb.lacunarity | float | octave frequency multiplier |
| perturb.persistence | float | octave signal decay |

## 6.8 fat glandular buffer noise parameters

| Name | Type | Notes |
|---|---|---|
| buffer.maxDeviation | float | maximum fraction of distance deviation |
| buffer.frequency | float | starting frequency |
| buffer.lacunarity | float | octave frequency multiplier |
| buffer.persistence | float | octave signal decay |

## 6.9 Voronoi segmentation variables

| Name | Type | Notes |
|---|---|---|
| voronoi.fatInFatSeedDensity | float (mm^-3) | fat voronoi seed density |
| voronoi.fatInGlandSeedDensity | float (mm^-3) | fat voronoi seed in glandular tissue density |
| voronoi.glandInGlandSeedDensity | float (mm^-3) | glandular voronoi seed density |
| voronoi.TDLUDeflectMax | float | maximum deflection (fraction of pi) |
| voronoi.minScaleLenTDLU | float | minimum length scale |
| voronoi.maxScaleLenTDLU | float | maximum length scale |
| voronoi.minScaleWidTDLU | float | minimum width scale |
| voronoi.maxScaleWidTDLU | float | maximum width scale |
| voronoi.minStrTDLU | float | minimum strength |
| voronoi.maxStrTDLU | float | maximum strength |
| voronoi.fatInFatDeflectMax | float | maximum deflection (fraction of pi) |
| voronoi.minScaleLenFatInFat | float | minimum length scale |
| voronoi.maxScaleLenFatInFat | float | maximum length scale |
| voronoi.minScaleWidFatInFat | float | minimum width scale |
| voronoi.maxScaleWidFatInFat | float | maximum width scale |
| voronoi.minStrFatInFat | float | minimum strength |
| voronoi.maxStrFatInFat | float | maximum strength |
| voronoi.fatInGlandDeflectMax | float | maximum deflection (fraction of pi) |
| voronoi.minScaleLenFatInGland | float | minimum length scale |
| voronoi.maxScaleLenFatInGland | float | maximum length scale |
| voronoi.minScaleWidFatInGland | float | minimum width scale |
| voronoi.maxScaleWidFatInGland | float | maximum width scale |
| voronoi.minStrFatInGland | float | minimum strength |
| voronoi.maxStrFatInGland | float | maximum strength |
| voronoi.glandInGlandDeflectMax | float | maximum deflection (fraction of pi) |
| voronoi.minScaleLenGlandInGland | float | minimum length scale |
| voronoi.maxScaleLenGlandInGland | float | maximum length scale |
| voronoi.minScaleWidGlandInGland | float | minimum width scale |
| voronoi.maxScaleWidGlandInGland | float | maximum width scale |
| voronoi.minStrGlandInGland | float | minimum strength |
| voronoi.maxStrGlandInGland | float | maximum strength |
| voronoi.seedRadius | float (mm) | check seeds in radius |

## 6.10 fat lobule parameters

| Name | Type | Notes |
|---|---|---|
| fat.minLobuleAxis | float (mm) | min lobule axis length |
| fat.maxLobuleAxis | float (mm) | max lobule axis length |
| fat.minAxialRatio | float | axial ratio min |
| fat.maxAxialRatio | float | axial ratio max |
| fat.minLobuleGap | float | minimum ligament separation between lobules |
| fat.maxCoeffStr | float | maximum of absolute value of Fourier coefficient as fraction of main radius |
| fat.minCoeffStr | float | minimum of absolute value of Fourier coefficient as fraction of main radius |
| fat.maxLobuleTry | integer | maximum number of trial lobules |

## 6.11 ductal tree parameters

| Name | Type | Notes |
|---|---|---|
| ductTree.maxBranch | integer | target number of branches |
| ductTree.maxGen | integer | maximum generation |
| ductTree.initRad | float (mm) | initial radius of tree |
| ductTree.nFillX | integer | number of voxels for tree density tracking |
| ductTree.nFillY | integer | number of voxels for tree density tracking |
| ductTree.nFillZ | integer | number of voxels for tree density tracking |

## 6.12 ductal branch parameters

| Name | Type | Notes |
|---|---|---|
| ductBr.minLen0 | float (mm) | minimum and maximum branch lengths per level |
| ductBr.maxLen0 | float (mm) | minimum and maximum branch lengths per level |
| ductBr.minLen1 | float (mm) | minimum and maximum branch lengths per level |
| ductBr.maxLen1 | float (mm) | minimum and maximum branch lengths per level |
| ductBr.minLen2 | float (mm) | minimum and maximum branch lengths per level |
| ductBr.maxLen2 | float (mm) | minimum and maximum branch lengths per level |
| ductBr.minLenDefault | float (mm) | minimum and maximum branch lengths per level |
| ductBr.maxLenDefault | float (mm) | minimum and maximum branch lengths per level |
| ductBr.maxChild | integer | maximum number of children |
| ductBr.childMinRad | float (mm) | minimum branch radius to have children |
| ductBr.childLevBound | integer | |
| ductBr.child00 | float | |
| ductBr.child01 | float | |
| ductBr.child02 | float | |
| ductBr.child10 | float | |
| ductBr.child11 | float | |
| ductBr.child12 | float | |
| ductBr.child20 | float | |
| ductBr.child21 | float | |
| ductBr.child22 | float | |
| ductBr.child30 | float | |
| ductBr.child31 | float | |
| ductBr.child32 | float | |
| ductBr.child40 | float | |
| ductBr.child41 | float | |
| ductBr.child42 | float | |
| ductBr.minRadFrac | float | minimum starting radius as a fraction of parent end radius |
| ductBr.maxRadFrac | float | maximum starting radius as a fraction of parent end radius |
| ductBr.radFrac0 | float | starting radius as fraction of parent end radius for first child |
| ductBr.minAngle | float | min angle between parent end direction and child start direction for children after first (fraction of pi radians) |
| ductBr.maxAngle | float | max angle between parent end direction and child start direction for children after first (fraction of pi radians) |

## 6.13 ductal segment parameters

| Name | Type | Notes |
|---|---|---|
| ductSeg.lengthBetaA | float | length distribution shape parameters |
| ductSeg.lengthBetaB | float | length distribution shape parameters |
| ductSeg.radiusBetaA | float | radius distribution shape parameters |
| ductSeg.radiusBetaB | float | radius distribution shape parameters |
| ductSeg.minLen | float (mm) | min and max segment length |
| ductSeg.maxLen | float (mm) | min and max segment length |
| ductSeg.maxCurvRad | float (mm) | maximum radius of curvature |
| ductSeg.maxCurvFrac | float | maximum length of segment based on curvature (fraction of pi radians) |
| duct-Seg.ductSeg.minEndRad | float | min and max end radius as fraction of start radius |
| ductSeg.maxEndRad | float | min and max end radius as fraction of start radius |
| ductSeg.angleWt | float | cost function preferential angle weighting |
| ductSeg.densityWt | float | cost function density weighting |
| ductSeg.numTry | integer | number of trial segments to generate |
| ductSeg.maxTry | integer | maximum number of segments to generate before giving up and reducing length |
| ductSeg.absMaxTry | integer | total number of segment tries before completely giving up |
| ductSeg.roiStep | float (mm) | step size for checking segment is valid |

## 6.14 vessel tree parameters

| Name | Type | Notes |
|---|---|---|
| vesselTree.maxBranch | integer | target number of branches |
| vesselTree.maxGen | integer | maximum generation |
| vesselTree.initRad | float (mm) | initial radius of tree |
| vesselTree.nFillX | integer | number of voxels for tree density tracking |
| vesselTree.nFillY | integer | number of voxels for tree density tracking |
| vesselTree.nFillZ | integer | number of voxels for tree density tracking |

## 6.15 vessel branch parameters

| Name | Type | Notes |
|---|---|---|
| vesselBr.minLen0 | float (mm) | minimum and maximum branch lengths per level |
| vesselBr.maxLen0 | float (mm) | minimum and maximum branch lengths per level |
| vesselBr.minLen1 | float (mm) | minimum and maximum branch lengths per level |
| vesselBr.maxLen1 | float (mm) | minimum and maximum branch lengths per level |
| vesselBr.minLen2 | float (mm) | minimum and maximum branch lengths per level |
| vesselBr.maxLen2 | float (mm) | minimum and maximum branch lengths per level |

Continued on next page

Table 4 – continued from previous page

| Name | Type | Notes |
|---|---|---|
| vesselBr.minLenDefault | float (mm) | minimum and maximum branch lengths per level |
| vesselBr.maxLenDefault | float (mm) | minimum and maximum branch lengths per level |
| vesselBr.maxChild | int | maximum number of children |
| vesselBr.childMinRad | float | minimum branch radius to have children (mm) |
| vesselBr.childLevBound | integer | |
| vesselBr.child00 | float | |
| vesselBr.child01 | float | |
| vesselBr.child02 | float | |
| vesselBr.child10 | float | |
| vesselBr.child11 | float | |
| vesselBr.child12 | float | |
| vesselBr.child20 | float | |
| vesselBr.child21 | float | |
| vesselBr.child22 | float | |
| vesselBr.child30 | float | |
| vesselBr.child31 | float | |
| vesselBr.child32 | float | |
| vesselBr.child40 | float | |
| vesselBr.child41 | float | |
| vesselBr.child42 | float | |
| vesselBr.minRadFrac | float | minimum starting radius as a fraction of parent end radius |
| vesselBr.maxRadFrac | float | maximum starting radius as a fraction of parent end radius |
| vesselBr.radFrac0 | float | starting radius as fraction of parent end radius for first child |
| vesselBr.minAngle | float | min angle between parent end direction and child start direction for children after first (fraction of pi radians) |
| vesselBr.maxAngle | float | max angle between parent end direction and child start direction for children after first (fraction of pi radians) |

## 6.16 vessel segment parameters

| Name | Type | Notes |
|---|---|---|
| vesselSeg.lengthBetaA | float | length distribution shape parameters |
| vesselSeg.lengthBetaB | float | length distribution shape parameters |
| vesselSeg.radiusBetaA | float | radius distribution shape parameters |
| vesselSeg.radiusBetaB | float | radius distribution shape parameters |
| vesselSeg.minLen | float (mm) | min and max segment length |
| vesselSeg.maxLen | float (mm) | min and max segment length |
| vesselSeg.maxCurvRad | float (mm) | maximum radius of curvature |
| vesselSeg.maxCurvFrac | float | maximum length of segment based on curvature (fraction of pi radians) |
| vesselSeg.minEndRad | float | min and max end radius as fraction of start radius |
| vesselSeg.maxEndRad | float | min and max end radius as fraction of start radius |
| vesselSeg.angleWt | float | cost function preferential angle weighting |
| vesselSeg.densityWt | float | cost function density weighting |
| vesselSeg.numTry | integer | number of trial segments to generate |
| vesselSeg.maxTry | integer | maximum number of segments to generate before giving up and reducing length |
| vesselSeg.absMaxTry | integer | total number of segment tries before completely giving up |
| vesselSeg.roiStep | float (mm) | step size for checking segment is valid |