**Mesh Creation**

See ‘D:\Depth electrode github example\Rat\_055\Mesh Creation’ on harddrive or ‘eit-nas/shared/Mayo Depth electrode github example\Rat\_055\Mesh creation’ on eit-nas for example of output from various steps

1. **Create file with position of depth electrodes for forward solution and mesher**

Overview

Finds the position of the centre of depth electrode contacts in mesh coordinate system. Based on these positions it creates a file with ‘fake’ electrodes defined in smaller intervals along each shank, which is required for refinement around electrodes in mesher (see section 5.3.1 MF\_phd\_thesis)

Code/Files required

1. electrodes\_pos\_bregma.m
2. Mesh\_tetra.mat (found in eit-nas or hard-drive)

Input required

1. Line 13-16 – change exp\_ML, exp\_DV and exp\_AP to coordinates of most anterior medial probe used in experiments (with respect to Bregma)

Output

1. *electrodes\_for\_depth\_RatXX\_3mm.txt*
   * Position file of centre of depth electrode contacts in mesh coordinate system
2. *electrodes\_for\_depth\_RatXX\_3mm\_small\_intervals.txt*
   * Position file of centre of ‘fake’ depth electrode contacts in mesh coordinate system

Notes

1. The way in which electrode coordinates are found is based on design of neural probe. If the geometry changes, code will have to change accordingly
2. The ‘fake’ electrodes for mesher only extend 2 mm in DV axis. This can be increased so that the whole shank up to surface of cortex is refined. This will, however, make the mesh much bigger.
3. **Generate Mesh**

Overview

Generate mesh with highly refined elements around depth electrodes and refined elements in a cuboid located in hemisphere of interest

Code/Files required

1. Use electroderefine branch on ‘EIT-team/Mesher **‘** on github
2. rat\_seg.inr (this doesn’t have dura) or rat\_seg\_dura.inr (this has dura) (these can be found in eit-nas/Mayo/Mesh or hardrive)
3. input\_idx.txt (example in EIT-team/Mesher/inputs)
4. electrodes\_for\_depth\_RatXX\_3mm\_small\_intervals.txt.

Input required

In input\_idx.txt

* 1. Section square refinement – change square\_centre\_x, square\_centre\_y, square\_centre\_z according to location of depth probes in mesh coordinate system (all values are in units of mm)
     + The centre of these coordinates was roughly based on the central location between the two neural probes. However, it was shifted a bit to cover the whole cortex where the cortical electrodes were placed.
  2. Section square refinement – change square\_x\_extent, square\_y\_extent, square\_z\_extent according extent of cuboid area that you want to refine
  3. Section square refinement - square\_cell\_size defines the size of the elements in this cuboid
     + This was set to 0.09 mm (90um) in my meshes
  4. Section standard refinement parameters – cell\_fine\_size, cell\_coarse\_size
     + These define the size of the elements in parts of the mesh that have not been specified (i.e close to the electrodes or in the cuboid)
     + Set cell\_fine\_size to same value as square\_cell\_size
     + Cell\_coarse size is size in the remainder of mesh. This was set to 0.5 (500um) in my meshes
  5. Section standard refinement parameters – electrode\_radius\_mm defines the radius around each electrode position that you want refined
     + This was set to 0.05 (50um) in my mesh. This cannot be less than the width of the depth electrode shanks as you then can’t remove them properly later
  6. Section standard refinement parameters – cell\_size\_electrodes defines the size of elements in the radius around the electrodes
     + This was set to 0.015 (15 um) in my mesh
  7. Section optimisation – lloyd, odt, exude, perturb are the optimisation methods undertaken to improve quality of mesh
     + When testing mesh to make sure elements in correct area are refined and position of depth electrodes are correct set all four of these to 0
     + Once you are happy with the refinement, set all four to 1 to generate final optimised mesh

Execution

1. A description of how to run the mesher is given in the README.md on the github (<https://github.com/EIT-team/Mesher>). This is the command that I ran from the directory ‘Mesher/’

./bin/mesher -i inputs/rat\_seg\_dura.inr -e inputs/electrodes\_for\_depth\_Rat\_XX\_3mm\_small\_intervals.txt -p inputs/input\_idx.txt -o Mesh\_Rat\_XX -d ./output/

* rat\_seg\_dura.inr: segmentation file
* electrodes\_for\_depth\_Rat\_XX\_3mm\_small\_intervals.txt: depth electrode position file
* input\_idx.txt: parameter file (described above)
* Mesh\_Rat\_XX: the name you want the save the output mesh

Output

1. If the mesher has run successfully in ‘Mesher/output’ there should be 5 files that have the name you defined for the mesh. The files of interest are Mesh\_Rat\_XX\_tetra.csv and Mesh\_Rat\_XX\_vertices.csv

Notes

1. The gradated refinement of elements between the electrodes and the fine elements in the cuboid and the fine elements and the coarse elements have been hard coded in ‘Mesher/src/Sizing\_fields.cpp’

* Lines 138 – 142 – Between the fine and coarse elements it has been defined to gradate the elements over a distance 1.5 times the extent in each axis
* Line 289 - Between the electrode elements and the fine elements it has been defined to gradate the elements over a distance 6\* electrode radius.
* These can be changed if you like. You will have to recompile the mesher if you do so. See README.md on the github (<https://github.com/EIT-team/Mesher>) for instructions of how to recompile in linux.

1. **Convert mesh to matlab format**

Overview

Reads in the .csv files output from the Mesher and puts them in the matlab format that is used for Meshes in the lab

Code/files required

1. convert\_mesh\_csv\_to\_mat.m
2. Mesh\_Rat\_XX\_tetra.csv and Mesh\_Rat\_XX\_vertices.csv
3. writeVTKcell.m

Input required

1. Line 4 – change name to output from Mesher

Output

1. Mesh\_Rat\_XX.mat

* Mesh.Nodes(:,1:3) – coordinates of all nodes in mesh
* Mesh.Tetra(:,1:4) – index of nodes that comprise each tetra element
* Mesh.Tetra(:,5) – conductivity index in each element (1 – CSF, 2 – Grey Matter, 3 – White Matter, 4 – Dura mater)

1. Mesh\_Rat\_XX.vtk

* vtk file of mesh so you can visualize in paraview

1. **Remove depth electrodes from mesh**

Overview

Based on the position of depth electrodes, elements that are contained within the shanks are removed

Code/Files required

1. remove\_depth\_electrodes.m
2. Mesh\_RatXX.mat
3. electrodes\_for\_depth\_Rat\_XX\_3mm.txt

Input required

1. Lines 1-3 - change names to point to correct files

Output

1. Mesh\_rods\_rat\_XX.mat
   * mesh with rod elements removed
2. rod\_ratXX.vtk
   * vtk file with elements contained within rods (useful to visualise in paraview to check they have been correctly removed)
3. Mesh\_rods\_rat\_XX.vtk
   * vtk file of mesh without rods

Notes

1. Rods removed are rectangular and extend 100 um in ML and 50 um in AP direction. In AP direction the rods removed are not centered around the electrode positions. They are removed so that the electrode positions lie on the surface removed.
2. **Create Hexahedral mesh**

Overview

Create hexahedral mesh for image reconstruction.

Code/Files required

1. minecraft\_mesh\_m.m
2. Mesh\_rods\_RatXX.mat
3. writeVTK\_cell\_hex.m

Input required

1. Lines 22 – 39 – change refine centre and refine extent to be around area where you want to make the hex mesh

Execution

1. In matlab terminal

[Mesh\_hex,k]=minecraft\_mesh\_m('D:\Rat\_056\Reconstruction\Mesh\Mesh\_rods\_Rat\_056.mat',200e-3,1);

200e-3 – size of cubic elements in mesh in mm

1 – whether or not to write VTK (see beginning of code for description)

Output

1. Mesh\_rods\_RatXX\_hex\_200um.mat
   * Hex mesh
2. Mesh\_rods\_RatXX\_hex\_test.vtk
   * vtk for visualization

Notes

1. The code is set up at the moment so it finds the index within the region specified and only makes hex mesh from this region. If you want whole mesh change line 52 to grey = Mesh.Tetra(ind\_el,:);
2. The Mesh\_hex.Nodes might come out in mm. Convert to metres to be consistent with tetra mesh