

## Delineating visual ROIs with SamSrf

SamSrf comes with tools that can be used to delineate visual areas directly in MatLab. These are located in the *SamSrf/Utils* folder. Since some time in 2021, there has been an auto-delineation tool that automatically draws some borders between regions based on an atlas. It requires a transformation from the *fsaverage* template brain based on an atlas. More and/or improved atlases may be added in the future. The auto-delineation is built into the DelineationTool (see below). However, you also run the auto-delineation by navigating into the folder containing your pRF data and running something a command like this in Matlab:

```
AutoDelineation('lh_pRF_Gaussian', '../surf', '../..fsaverage/surf')
```

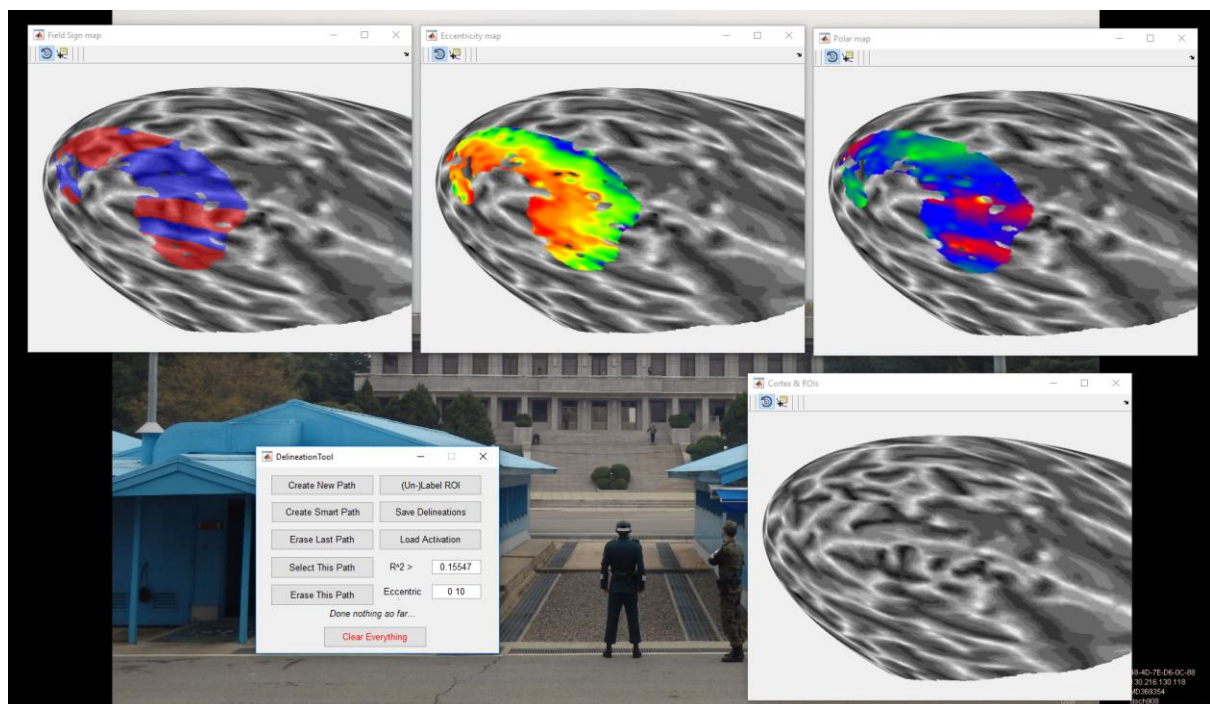
The first input is the name of the pRF data file (this could also be a bilateral file containing both hemispheres – in this case the tool will analyse each hemisphere separately). The second input is the path to the subject's *surf* folder. The third input is the path to the *surf* folder of the *fsaverage* template. Optional inputs allow you to choose the atlas to use and other parameters of the map.

The auto-delineation tool then places some initial borders on your retinotopic map derived from the template brain. Then, it iteratively moves around these borders to find the polar angle reversals and the iso-eccentricity lines. This is then saved as a delineation file. This auto-delineation saves a lot of time but it is by no means perfect. You need to use the manual delineation tool to correct and adjust these maps, close open borders, and also to label the individual ROIs. Obviously, you can simply skip the auto-delineation altogether and do it all manually. To run the manual delineation you call:

DelineationTool

This will ask you to select the pRF mapping data file (e.g. *lh\_pRF\_Gaussian.mat*). It will automatically determine the occipital ROI, the kind of surface to render, the surface files, the p-value to determine your starting  $R^2$  threshold, the list of possible ROIs to delineate, and the thresholds. You can change the defaults for any of these things by editing *DelineationTool.m* directly. Look for the lines of code indicated by '%%%' '%%%' '%%%' '%%%' '%%%'. However, except maybe for the threshold, for most people this won't be necessary. Note that the list of ROIs is predetermined by the atlas used for auto-delineation, if this is what you did.

The delineation tool will open four windows with spherical (or others, if you changed it) meshes and a menu interface. You can arrange the windows on the screen however you please:





\* Please note that in later versions we changed the default colour schemes from what is shown here.

The four maps\* are from left to right: the *field sign*, telling you whether a map is an upside-down (red) or normal (blue) representation, the *eccentricity* map, and the *polar* map. The empty map in the bottom right will contain the paths you drew and, if you labelled any ROIs, the colours of the ROI labels. You will also notice that the camera angle is the same for all maps. When you rotate one map or draw a path, it will also update those features in the other three maps. Any paths you draw are saved in a file called something like *del\_lh\_pRF\_Gaussian.mat*. Any ROIs you label will be saved in a new subfolder called something like *ROIs\_pRF\_Gaussian*.

Note: in later SamSrf versions the field sign is no longer computed automatically. This requires smoothing the final map quite heavily and we have opted for less smoothing in the analysis. You can generate the field sign map if you want but it often doesn't really help that much. If no field sign is present in your map file, the delineation tool now instead shows the  $R^2$  map as this can tell you a little about artifacts.

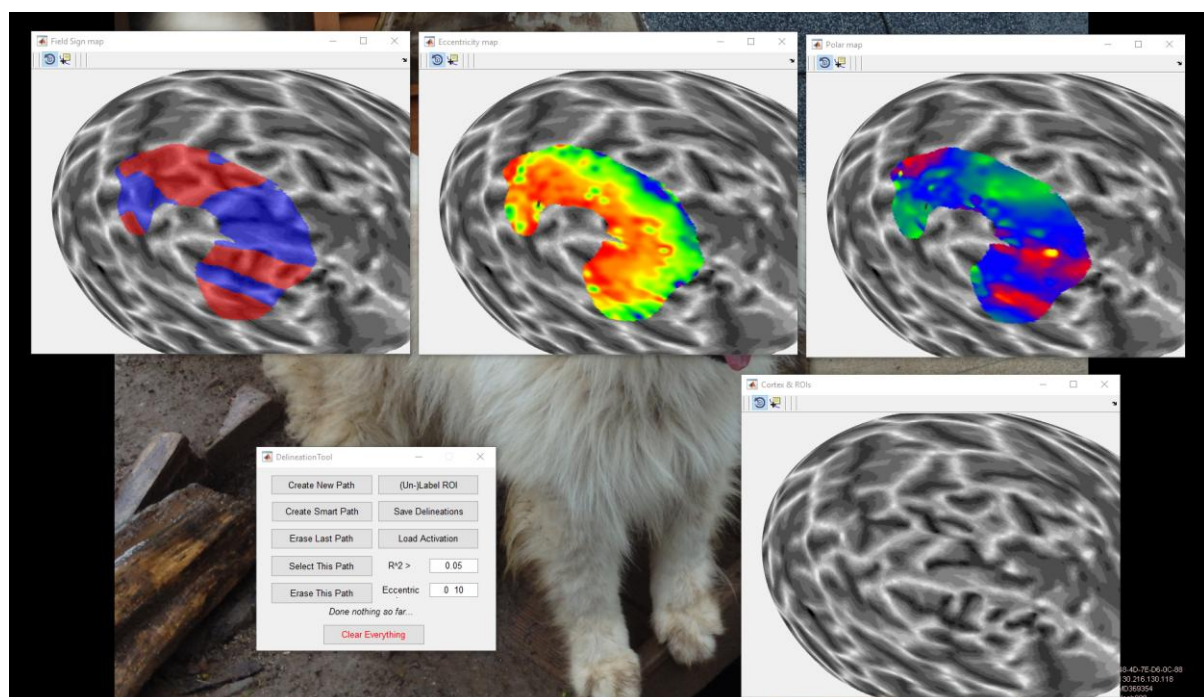
The GUI contains a number of buttons. The exact appearance of the GUI may not be the same as shown here as we move buttons around etc based on whether people report back that there are more user-friendly locations. We have also changed the colour palettes used in SamSrf to be more amenable to people with atypical colour vision. But the principle should stay the same.

Each map window has two tools in the toolbar at the top:

- the rotation tool 
- the data cursor tool 

The former is selected by default and allows you to rotate the mesh in three dimensions. The data cursor tool is used for selecting vertices, either for placing waypoints for paths or for selecting a ROI for flood filling a label. It is advisable to select the data cursor tool in one or two windows and leave the rotation tool selected in the others. This way you can easily switch between both modes. Ideally, you need the data cursor tool in the *polar* map and perhaps also the *field sign* map.

To begin we rotate the maps so I have a good straight view of V1, that is, the region in between the red and green right inside the calcarine:



\* Please note that in later versions we changed the default colour schemes from what is shown here.

Before we begin, we might want to threshold the maps. As mentioned above, the default R2 threshold is defined by a p-value in the code and out of the box this is  $p < 10^{-10}$ . However, you can adjust this threshold in the  $R^2 >$  box any way you please. In the maps shown above I dropped the threshold to 0.05 which fills in some of the little holes in the map.

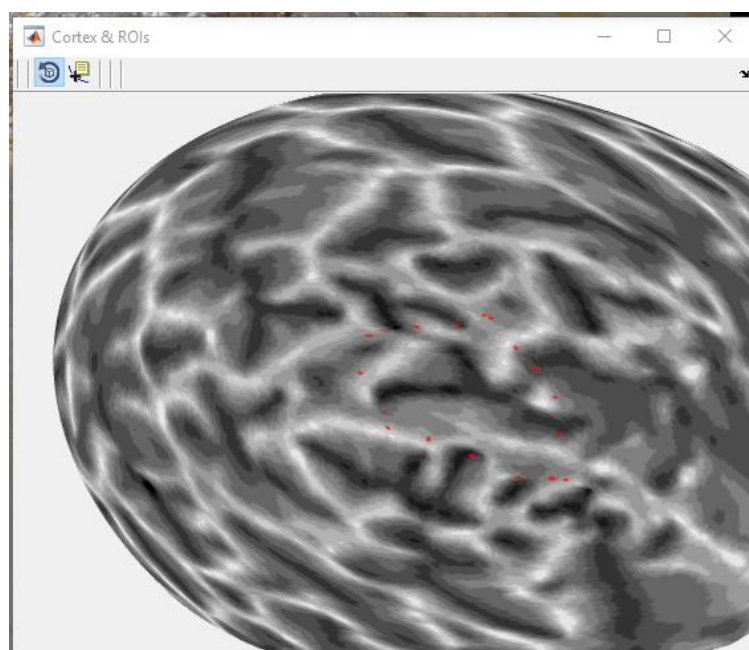
You can also change the *Eccentricity* range to only show pRFs within those eccentricities. This can be advisable because pRFs in the very centre or outside the stimulated part of the visual field tend to be noisy due to off-the-edge artifacts. If you used the *samsrf\_surfcals* function to post-process your pRF map and calculate a *field sign* map, then your map will already be restricted to an eccentricity range and many artifacts will have been removed already. In the above maps, I left it at the default.

So far so good. I now select the data cursor mode for the polar map and start drawing. For those familiar with *tksurfer* delineations, this isn't new. Simple put a string of points along the path you want to draw. The tool however forces you to place your points fairly close together to avoid errors. Whenever a point is selected correctly, a small red dot will appear in the empty map in the bottom window. Whenever it is not selected because it is too far from the previous waypoint (which could be because of an error), then *no red dot will appear but there will be a warning in the Matlab command window*.

**It is important to note that whenever you rotate the mesh, the waypoints get deleted!** This is to ensure that no erroneous waypoints get created all the time. This is different from *tksurfer*. If you created a set of waypoints and don't want to lose them but need to rotate the mesh, you should draw the path for them first.

You may also get an occasional error message inside the display window itself. These are ugly MatLab bugs that I have so far had no luck in killing. They shouldn't be much of a worry though. You can usually just ignore them. If they are too bad, or if these messages appear all the time which I heard happens on some Linux distros, then you can change the way these messages are displayed. There is a line in the code for *DelineationTool.m* that is commented out by default. You need to uncomment this to display messages in the corner of the window. At the point of writing this tutorial the relevant line is 209 but it is marked by a message "THIS LINE NEEDS TO BE UNCOMMENTED..."

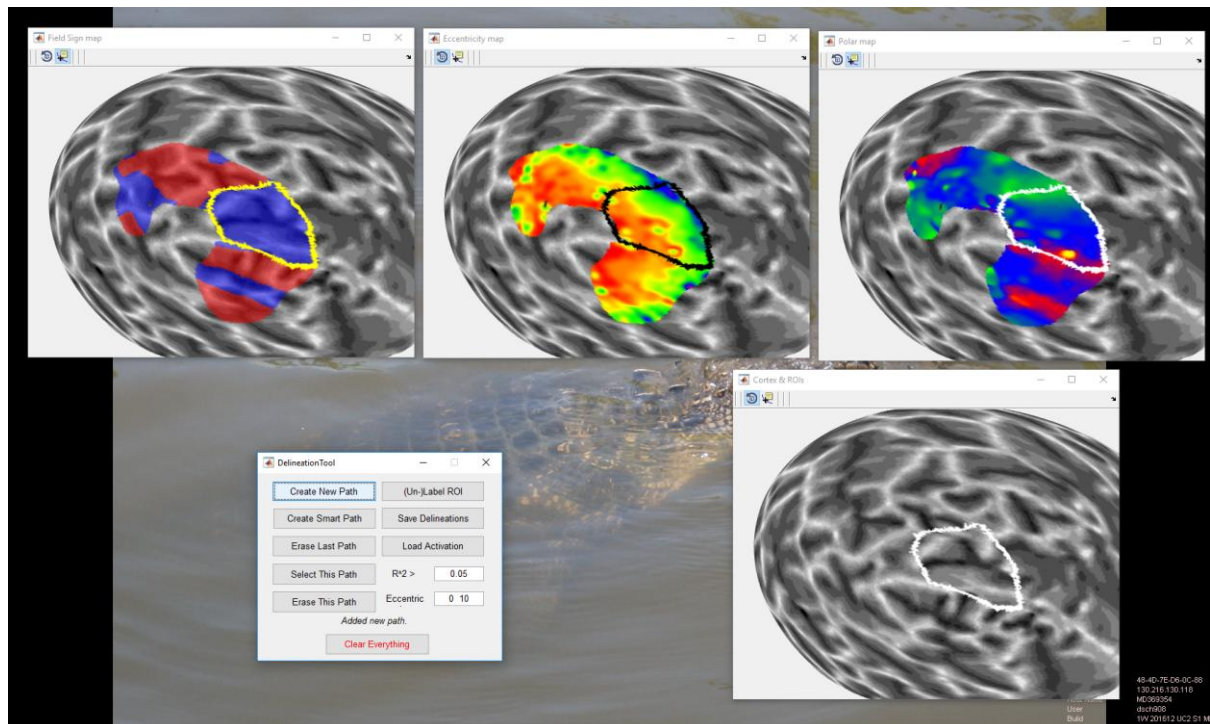
Below I show a set of waypoints selected along a path surrounding V1.



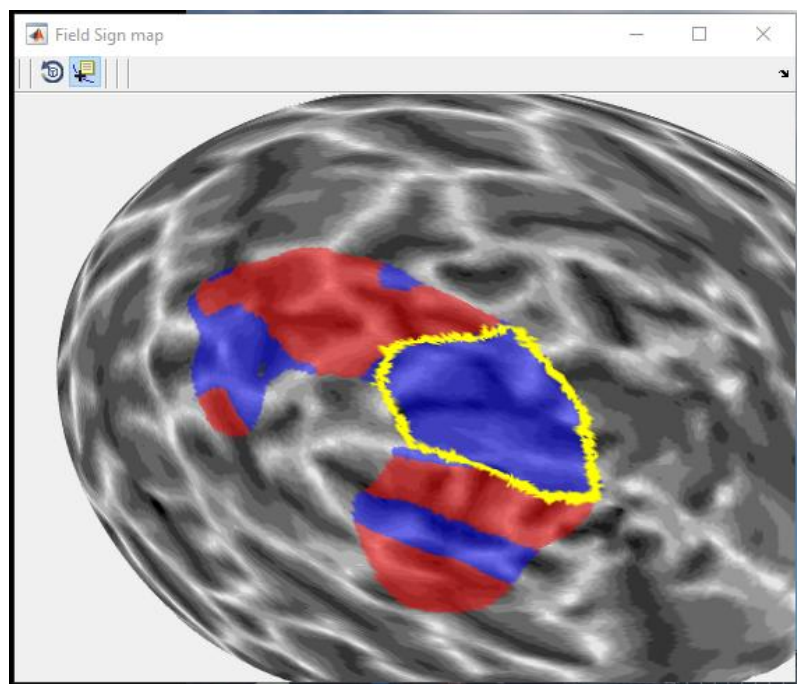
If you are happy with that path selection and there are no erroneous red dots anywhere, you can click **Create New Path** in the interface. This will draw paths along these waypoints which will appear in all of the four windows.

\* Please note that in later versions we changed the default colour schemes from what is shown here.

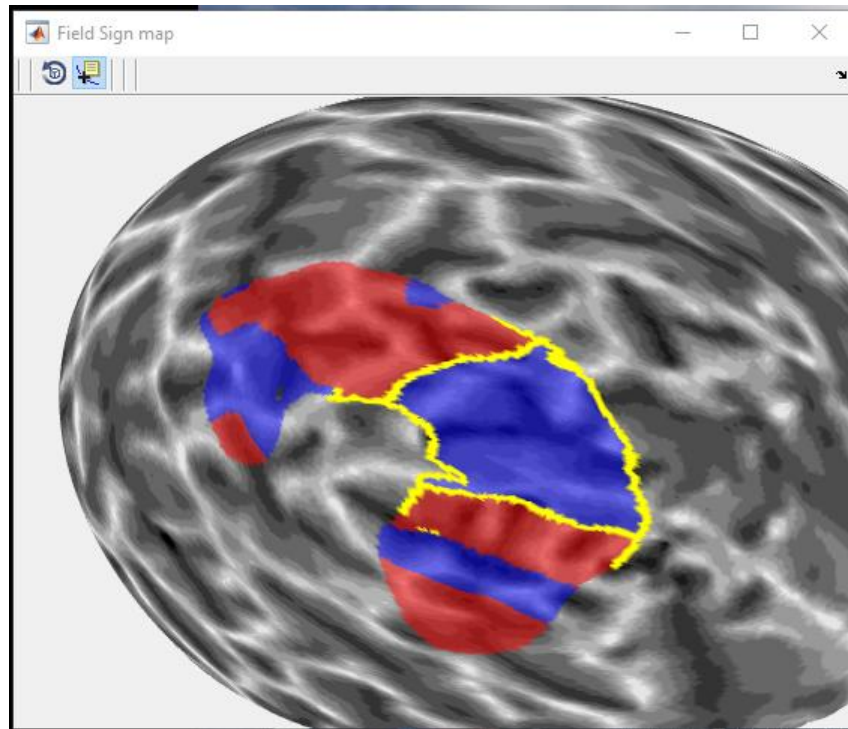




As you can see, the path nicely surround V1. I drew it along the middle of the red and green stripes in the polar map and the edges of the structured map – but you can see it also nicely surrounds the red area in the *field sign* map although my lines were not perfect in following the field sign reversals:



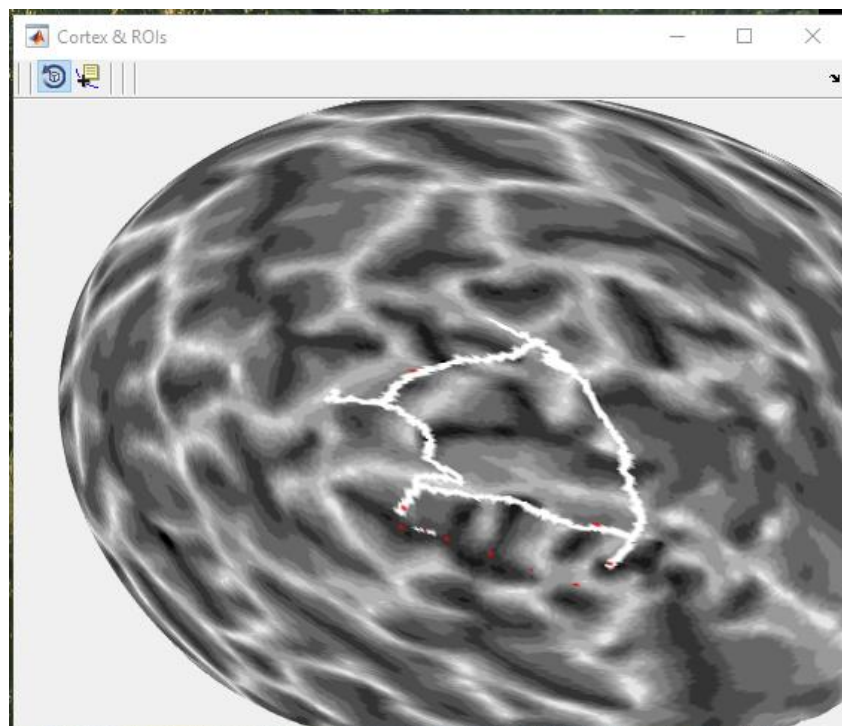
Perhaps I should have used the *field sign* map for drawing the borders instead. The **Create Smart Path** button allows you do to just that: Instead of simply connecting the waypoints you selected, it uses each waypoint as a centre for a small circular region of interest. It searches these regions for any vertices that contain either a field sign reversal or that are at the edge of the thresholded map. This feature can massively speed up your delineation, but it must be used wisely.



**You need a high-quality *field sign* map and you must be careful to still manually delineate regions where the *field sign* map fails.** In the below example, the field sign map is great for V1 and the ventral regions, but it clearly failed for V2d and V3d. So you will need to draw paths there manually and use some personal judgement (see tutorial below about how to delineate the regions).

*Note, that if it seems impossible to select a new waypoint, or the error messages don't want to go away, it may be time to rotate to a better angle. MatLab's vertex selection functionality is far from perfect. Sometimes it seems to accidentally select vertices behind the surface you click on which are on the far side of the mesh.*

I continue to do the same for V2v:

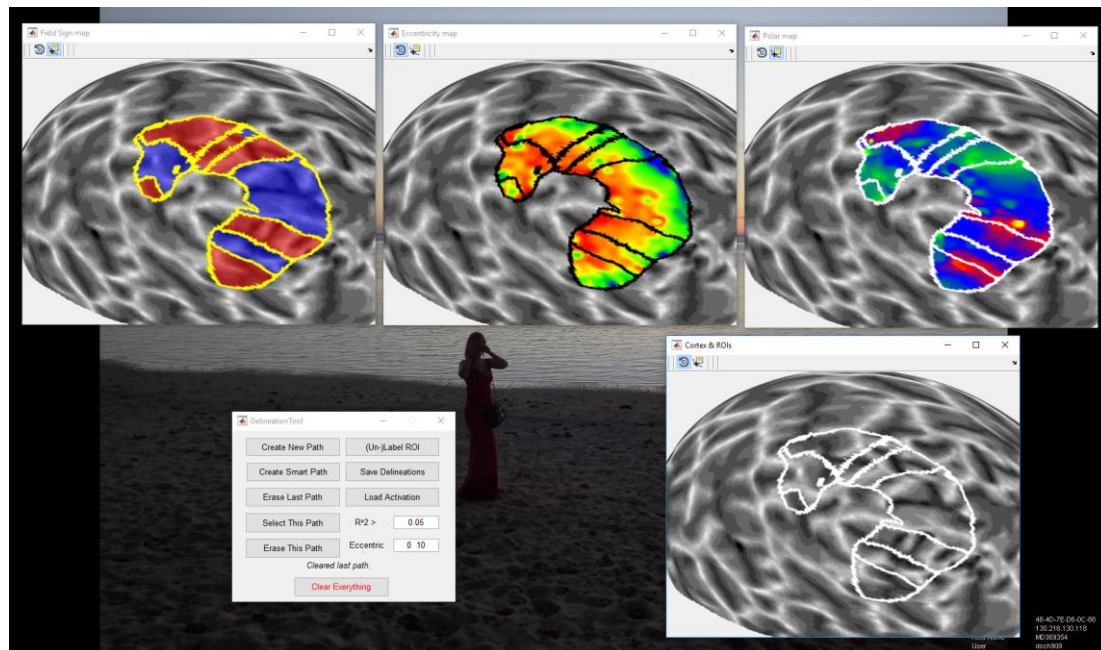


As you see, I now simply connect the new path to the one I already drew. There is no need to close it up. This, again, is just like in *tksurfer*. The paths are wider than in *tksurfer* though. They are usually 3

\* Please note that in later versions we changed the default colour schemes from what is shown here.

vertices across. When flood-filling a ROI it will include the outer ones but there may be vertices in the middle that may remain unlabelled. This helps preclude partial volume effects although by and large it should be similar to the labels you produce in *tksurfer*.

But let's not get ahead of ourselves. I continue drawing and adding paths until I delineated the borders of all the visual regions I can make out.

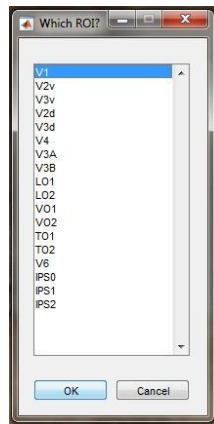


I've delineated V1-V4, V3A and V3B here. Let's leave it at that for now. V3B is already shaky in this map and I also screwed up the V3-V3A border a bit (Follow what I say, not what I've done...). Let's assume I want to label these ROIs. The first thing you want to do is click **Save Delineations** in the interface. This saves the *del\_lh\_pRF\_Gaussian.mat* file with the paths, so next time you run the delineation tool they will be loaded (you can also load them in the rendering functions in the SamSrf user interface).

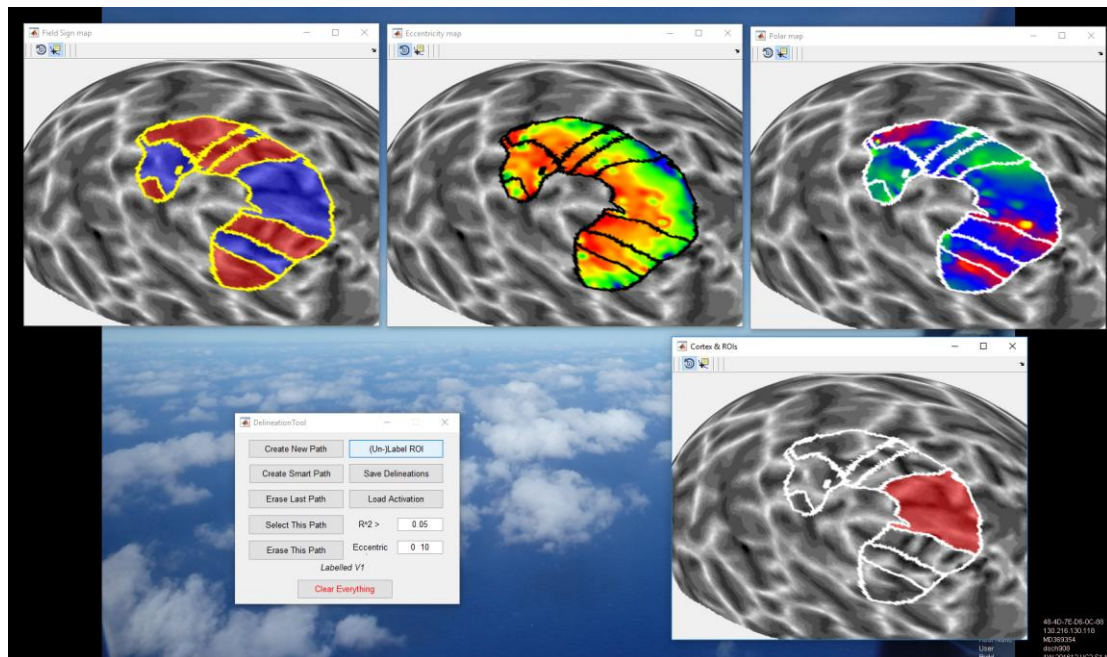
However, paths alone aren't very useful. What we need for analyses and other experiments is ROI labels, so the next thing we want to do is label the ROIs. To do so you should first rotate the map to remove any residual waypoints that might have been left. I would now put all the three maps on rotation mode and the empty map on the bottom into data cursor mode – but this is really up to you. I just find it more convenient that way.

To select a ROI you simply click on a vertex inside the ROI. In our example I clicked inside V1 (if you don't know where that is have a look at the tutorial below or ask someone). Now, to label all the vertices inside the ROI you click on **(Un)-Label ROI**. This opens a list dialog with all the possible ROI labels you might want to choose from (you can modify this for your own purposes by changing the defaults). Simply select V1 from the list and click OK:

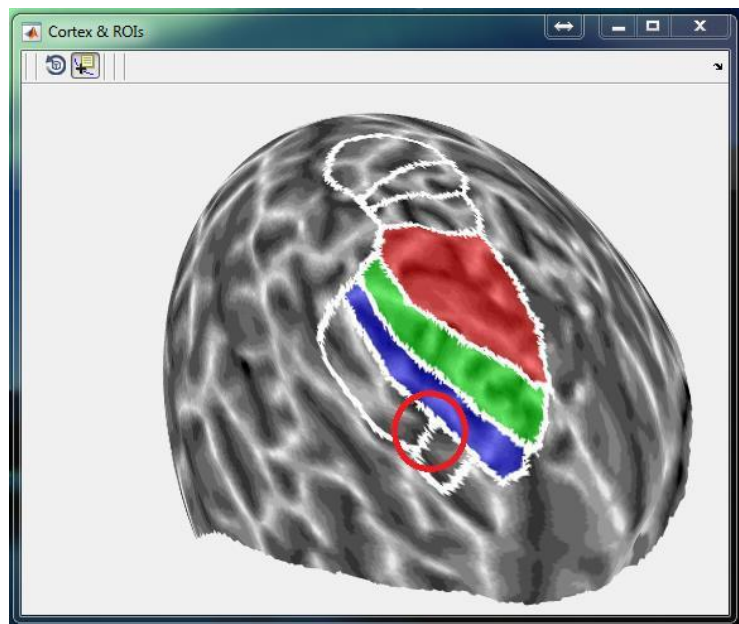




Lo and behold, V1 is now labelled in red. At the same time a label called *lh\_V1.label* has automatically been saved in the *ROIs\_pRF\_Gaussian* folder. If you open the *(Un-)Label ROI* dialog again, you will now also see a number next to V1. This tells you the number of the vertex (not the number of vertices!) that you selected and that was used to seed the filling algorithm. So if there is a number in that list you know that the program believes V1 has been labelled.

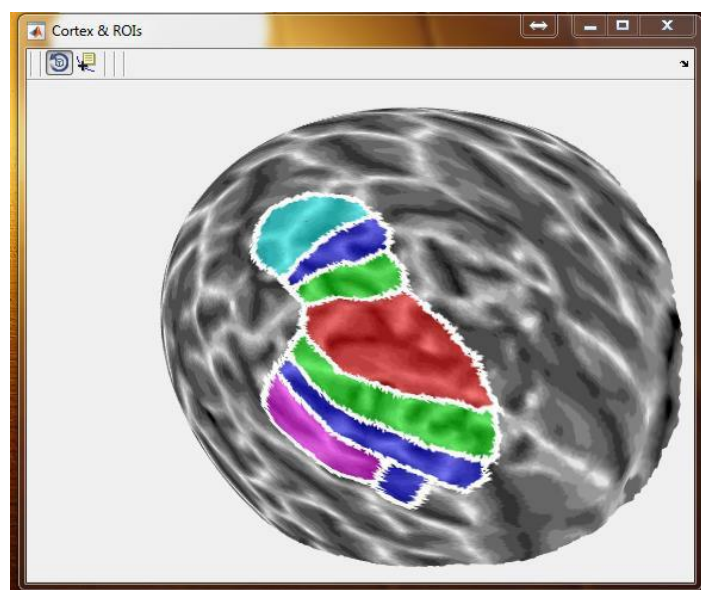


We continue this procedure for the other ROIs (V2 and V3 are separated by ventral and dorsal quadrants). **Watch out for any potential holes or gaps in the paths!** People who used *tksurfer* will be familiar with that problem. Unfortunately this is difficult to avoid completely. For example, in this image (of a different map) there is a gap in the border between V4 and VO1 (see red circle):



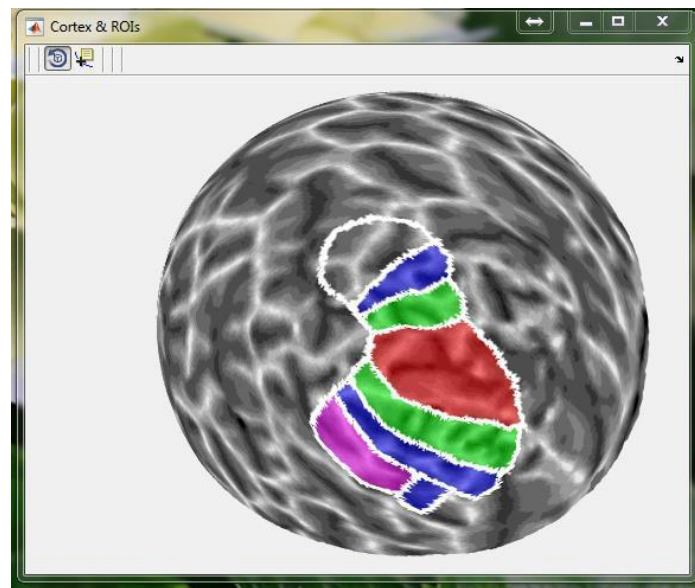
To fix this, simply try to draw another tiny path filling the gap. Then just continue as before filling in all the ROI labels. *If you find that the ROI labelling dialog is really slow, you may switch the data cursor mode to another window (Aren't MatLab bugs wonderful?...)*

If you find you made a mistake by labelling a ROI with the wrong name, you can easily unlabel it. In this image I accidentally labelled V3A as V6. (You can tell by its cyan colour – V3A would be red):



You can use the **(Un-)Label ROI** button to unlabel the ROI. **Simply unselect all vertices and then pick V6 in the list!** (You unselect vertices by first rotating the meshes to clear any waypoints). This will remove the number of the vertex index from that ROI in the *(Un-)Label ROI* dialog.





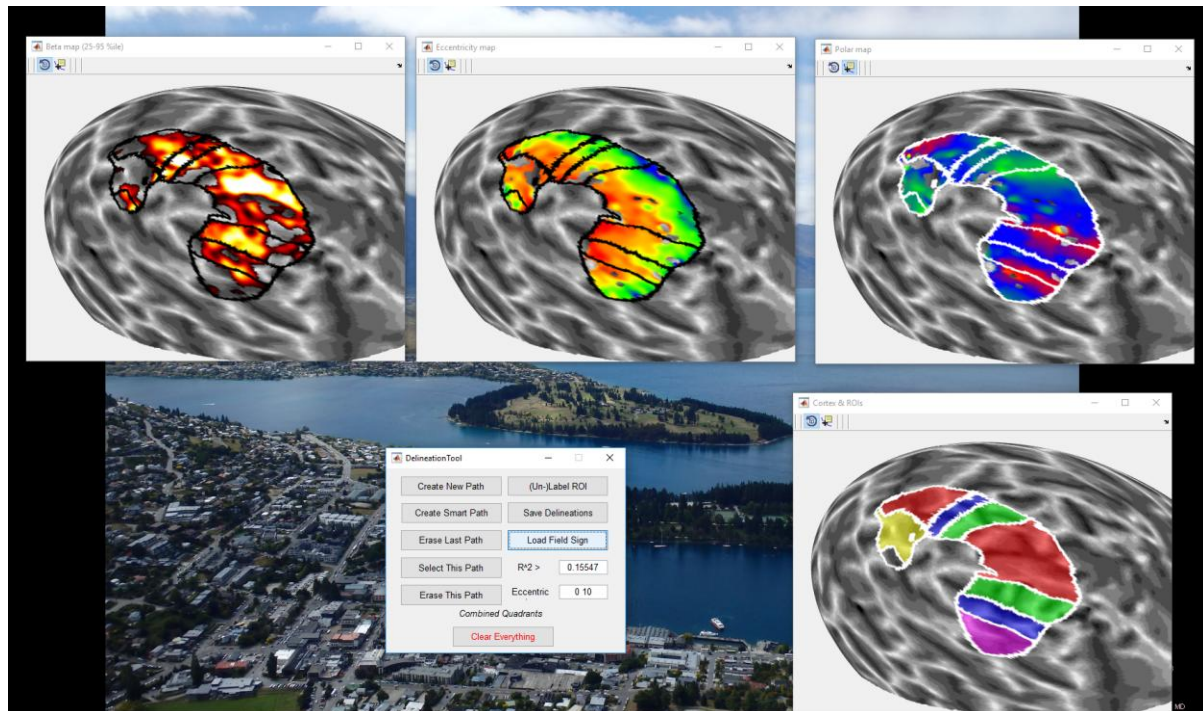
Now I can select it again and label it as V3A. **Importantly, when you unlabel a ROI, the associated ROI label will also be deleted from the disc!** This shouldn't cause any problems but you should be aware of this.

When all the ROIs have been labelled according to your wishes, you should click the **Save Delineations** button again. This way also the ROI assignments will be preserved. Since V2 and V3 are separated into the ventral and dorsal quadrants but we usually want to analyse those areas as complete hemifield representations, the **Save Delineations** button will try to combine them. If one or both quadrant labels don't exist yet, it will give a warning in the MATLAB command window.

The function of the other buttons is mostly self-explanatory: **Erase Last Path** removes the last path that was drawn. **Select This Path** will select the path you clicked on most recently and highlight it. You can then use **Erase This Path** to get rid of it. You can thus clear paths either in the reverse order they were created or by selecting them manually. Finally, **Clear Everything** wipes the maps clean of any paths or ROI labels.

The delineation tool also allows you to load simple activation maps. There is a new button in the menu (not shown in the previous screenshots) that reads **Load Activation**. Clicking this will open a dialog for selecting a Srf file with the activation map (e.g. a GLM contrast map). If there is more than one map (i.e. volume) in that *Srf.Data* variable it will open a second dialog to select the map you want. This map will be thresholded in exactly the same way as the rest of the map (i.e. with  $R^2$  threshold and eccentricity range).

This then replaces the field sign map with the activation map. Further thresholding and clipping of the activation map is determined by the percentiles defined in *DelineationTool.m*. The default is to show activations within the 25<sup>th</sup> and 95<sup>th</sup> percentile. This feature is to allow you to compare your retinotopic maps with other activation maps when delineating ROIs so it will be restricted to the same activation patterns as the retinotopic map. If you want to display maps side by side, it is recommended that you use the *DisplayMaps.m* tool instead.



When the activation map has been loaded the button in the menu changes to **Load Field Sign**. Clicking this will restore the field sign map again.

## Delineation guide for visual areas

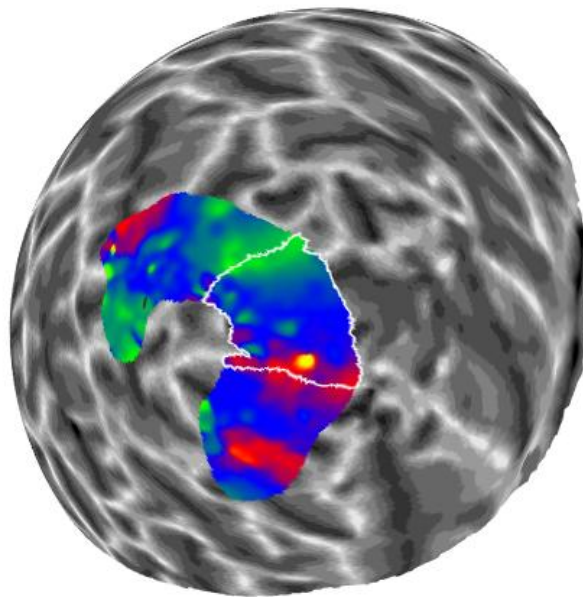
Delineating the earlier visual areas V1-V3 is usually quite straightforward, but even with them there are often tricky cases and open questions. Many higher areas are more confusing. Here I try to give some guidance as to how I have been delineating these cases. For a basic background you may also enjoy reading my simple (and now very old) retinotopy tutorial for phase encoded analysis:

<https://doi.org/10.6084/m9.figshare.1513839.v1>

More special cases may be added in future versions of this tutorial. Please let me know if you have any questions, comments, or corrections about the advice given here:

### V1

V1 contains a full hemifield map and is located fairly accurately within the calcarine sulcus. Thus it should span from a green stripe on the cuneus (dorsal bank of calcarine) through blue in the depth of the calcarine to a red stripe on the lingual gyrus (ventral bank).



Note that the red/green stripes may not always be continuous. In this example (which I would say is actually a pretty good map) the red stripe on the ventral side is disrupted by blue near the occipital pole. Such interruptions are very common and likely artifacts of the surface sampling of the functional voxels. In worse cases the colour may even be green (which is much more worrying but fortunately fairly rare). You need to use your common sense a bit. Borders between regions don't just zigzag erratically around the brain. Changing the  $R^2$  threshold may help remove rubbish data. Moreover, according to the theory that mechanical tension due to intracortical connectivity causes the cortical folding, the vertical meridians (red and green stripes) should coincide with the crest of the gyri. Especially in V1 this is typically the case and this is why anatomical predictions of retinotopic maps work so well. You can see this here somewhat although the correspondence isn't perfect.

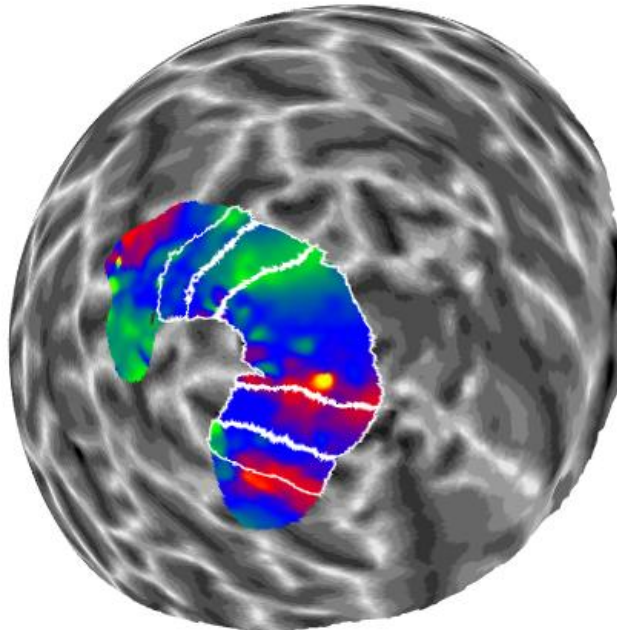
### V2 and V3

V2 and V3 are both segregated into two quadrant field maps, one in ventral cortex and one on in dorsal cortex. So V2d runs from the green stripe of the V1 border to the middle of the blue stripe. V3d

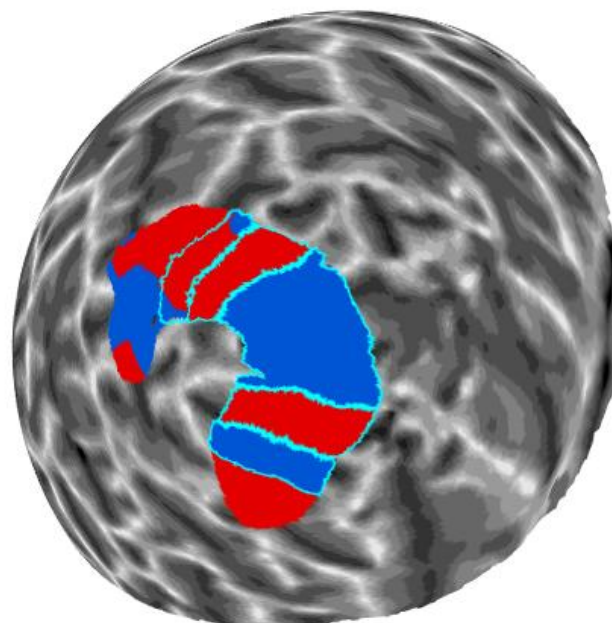
\* Please note that in later versions we changed the default colour schemes from what is shown here.



then runs from that blue stripe to the next green stripe. Conversely, V2v runs from the red stripe of the V1 border to the blue stripe, and V3v runs from that blue strip to the next red stripe.



In the example above, the ventral borders are pretty clear. However, the dorsal borders are trickier. The V2d-V3d border is disrupted by a washed out bluish green and the outer border of V3d is interrupted by a large blue gap. This was the example from the tutorial above about using the *DelineationTool*. For the most part, I drew these borders using the Create Smart Path feature so they closely follow the field sign reversals. The dorsal borders between V2d and V3d I drew manually but I used the field sign blobs on either end as a guide as to where these areas should be:



I simply continued the trajectory of the little blue blob on the peripheral edge to tell me the likely borders of V3d to gradually converge towards the foveal confluence. For the outer border of V3d I also used the polar map to connect between those green islands. **Critically, you want to go for the peak of the (in this case, green) colour.** This all seems fairly reasonable to me but as you can see cases like this requires a judgement call.

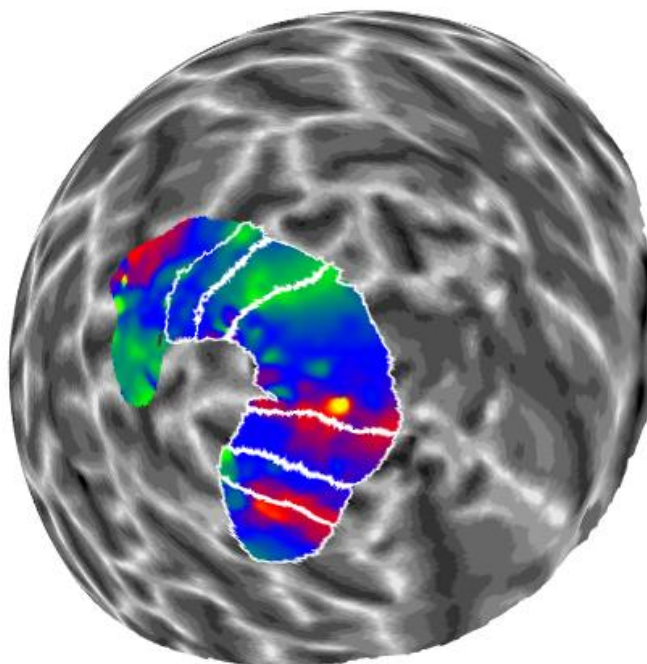
It's hard to know what the correct answer is here. This scenario, a washed out green regions where V2d and V3d should be, is surprisingly common. Again the field sign and eccentricity maps will help a

\* Please note that in later versions we changed the default colour schemes from what is shown here.

little here because it tells you how far the occipital fovea extends (which tells you how far V3d should go) and it may also segregate the upside-down map of V3d from the normal map in V2d. It is also important to use the presence of V3A as a guide. In these maps we clearly have a full hemifield from green to blue to red indicating V3A (see below). This tells you that anything posterior to V3A is presumably V3d.

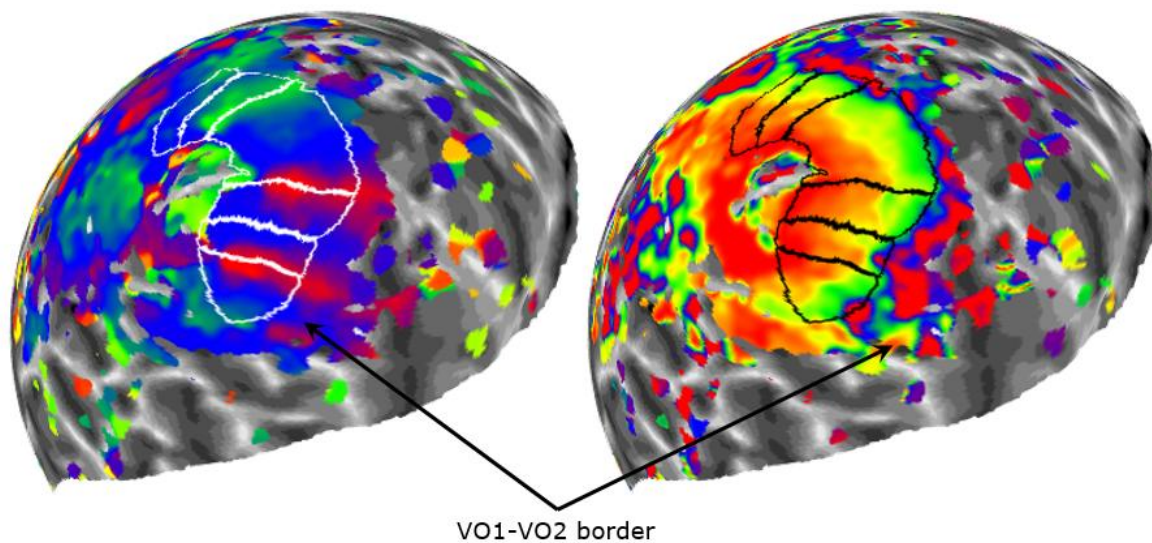
## V4

Some people call this hV4 to distinguish it from monkey V4, which appears to be organised differently. Whatever you call it, based on my own experience at least, human V4 always contains a hemifield map of the contralateral visual field. So it should run from the red stripe at the outer border of V3v to a green stripe. This green stripe is often disrupted by blue blobs – but in my experience there is almost always at least a bit of green, especially towards the more peripheral parts of V4.



In fact, this green peripheral bit indicates the peripheral border of V4. This is a polar angle reversal around a lower vertical meridian (just like the border between V1 and V2d). At this point the green outer border of V4 (that is, the one that is parallel to the border with V3v) makes a 90 degree turn towards V3v. Anything beyond this green reversal along what would be more peripheral V4, if V4 behaved like V1-V3, are the VO areas. They are VO1 and VO2 and possibly more. VO1 and VO2 contain a full hemifield as well and they share a fovea that is *separate from the occipital fovea*.

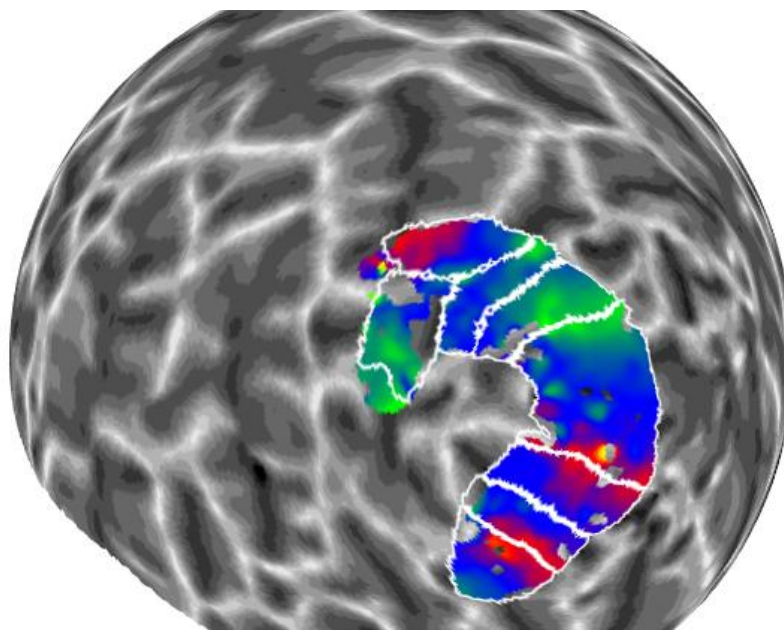
All of this should be visible in the maps below. This is from the same participant but a different analysis and using different thresholds. I didn't draw labels for VO1 and VO2 (because my labels are based on the smooth field sign map where I didn't see these) but I indicated what seems to be the border by arrows. This location corresponds to an upper vertical meridian (red polar angle) and a foveal representation (red in eccentricity map). In the best case scenario you should however be able to distinguish them via the field sign also: VO1 should be blue (mirror map) and VO2 red (normal map).



If you paid attention, this means that V4 is heavily biased towards the central visual field because it either doesn't represent the periphery at all or in an extremely compressed way. This arrangement is similar to the relationship between V3d and V6 – but since I have no maps that show that I won't show you any examples of that for now...

### V3A and V3B

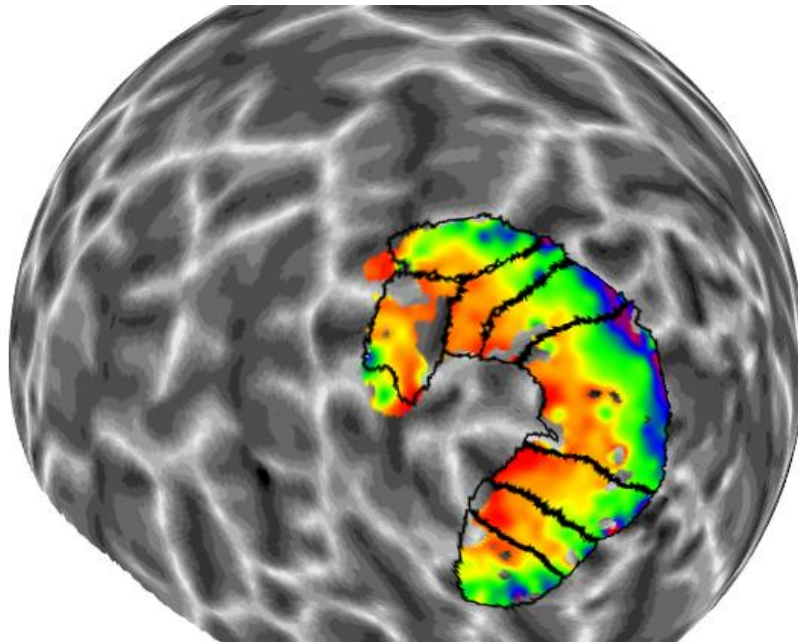
V3A is probably the most easily recognisable area after V1. It contains a full hemifield and is dorsal and anterior to V3d. Even in awful maps you usually see a hint of the red border characterising the anterior extent of V3A. Next to V3A, on the side pointing towards the occipital pole, you will find V3B, a rather peculiar area that nobody seems to quite agree on.



In really good maps, you will see that the outer red border of V3A suddenly takes a sharp 90 degree turn veering away from the occipital pole in an anterior-ish direction. This feature indicates one border of V3B. The other border should be a green reversal.



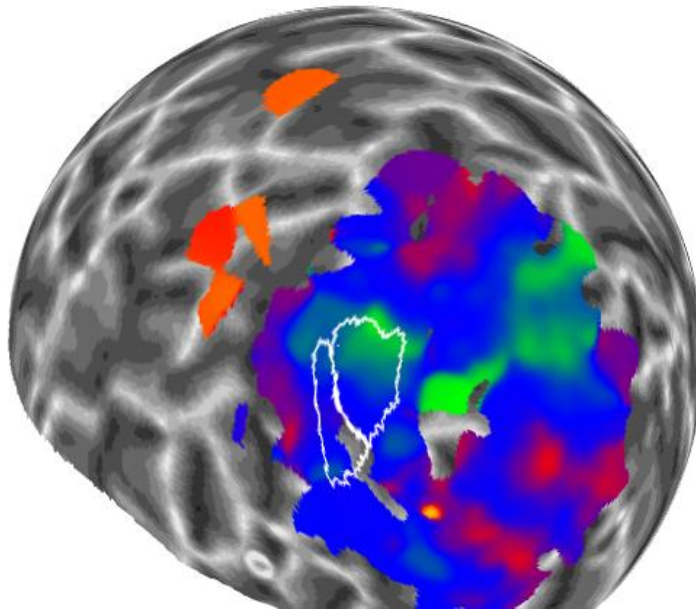
Critically, V3A and V3B share a common fovea which is usually different from the occipital fovea, even though the two foveas are usually attached. In the example below it looks as though it is the occipital fovea. In other brains, the distinction is more obvious because the eccentricity gradient makes a sharp turn out there.



### LO areas

LO is composed of (at least) LO1 and LO2. A poster (possibly a publication too by this point?) by the York group claimed that there are also LO3-LO6, which are located more laterally in the region that many traditional localiser types call LO and/or OFA (in fact, they suggested that LO3-LO6 *are* OFA). You often need a lot of imagination to see the LOs and I can't honestly say I have made out any maps beyond LO1 and LO2.

LO1 and LO2 are supposed to contain full hemifield maps. So the border between them should be a red stripe (upper vertical meridian) running towards the occipital fovea, which both these regions share. On the other side both LO1 and LO2, should be bounded by a green stripe (lower vertical meridian). I have rarely seen this arrangement. Much more likely it looks as in the example below, where we have mainly blue (horizontal meridian and at best a tiny purple-red spot. This is where I drew the border between LO1 and LO2.

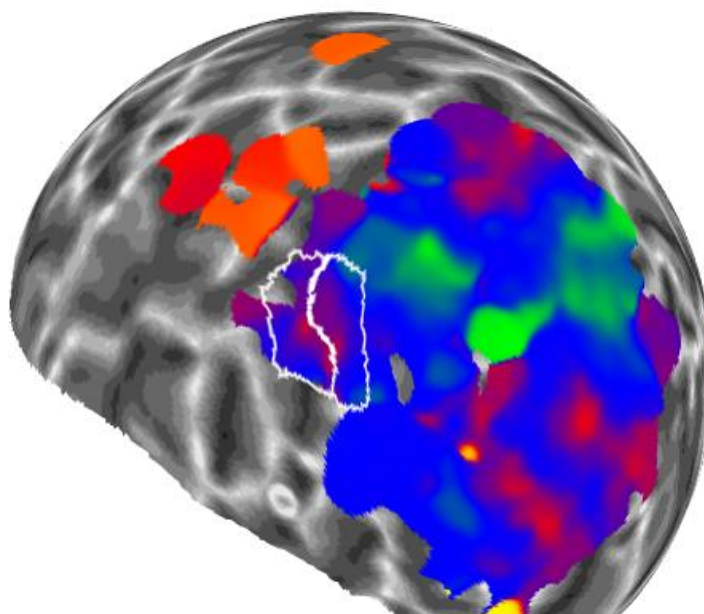


You'll notice that in these maps I dropped the threshold to show a lot more. The strictly thresholded and eccentricity-bounded maps I showed before wouldn't really show anything here. I also removed the earlier areas here to avoid confusion. The blue outer border of LO2 should really be green (and there was a hint of green here) because LO2 is supposed to be a full hemifield.

### **MT+ complex (TO areas)**

This brings us to the MT+ complex. People of the original London school of thought may choose to call this V5, in honour of Semir Zeki's naming of that area (and so you also often see the rather awkward V5/MT in the literature). The important thing to keep in mind here is that MT is only a small area that is part of the motion sensitive complex in the medial temporal cortex.

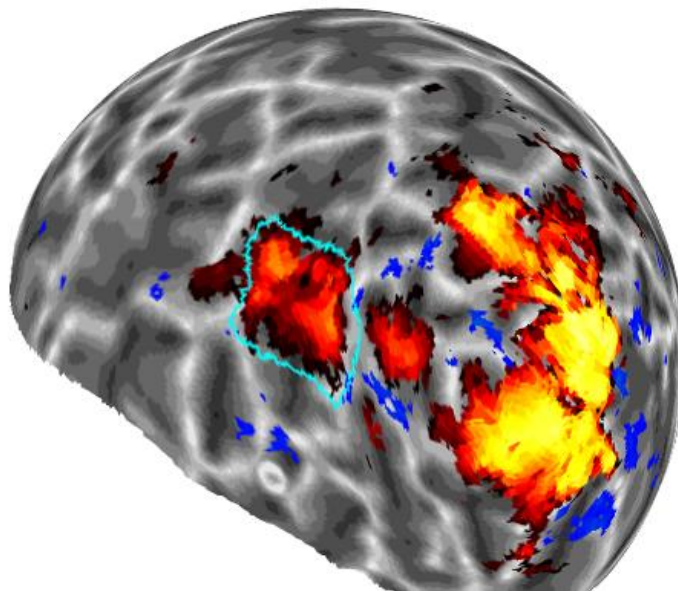
According to the Wandellian nomenclature, there are actually at least two retinotopic regions here, anterior to LO2. They are once again full hemifield representations although you may not see the lower vertical meridians (green stripes). You do tend to see the upper vertical meridian (red stripe) where the field sign for those areas reverses. Going from posterior to anterior they named these two areas TO1 and TO2.



It has also been shown that TO1 is the actual classical area MT (or hMT, to distinguish it from the monkey literature). TO2 is probably area MST which supposedly has somewhat different response properties. These two areas also have a separate foveal representation although that can be difficult to make out. There may be more areas that are part of this cluster. If you ask Marty Sereno, he will tell you that there are lots of small areas in this part of cortex (similar to the reports about the LO areas mentioned earlier).

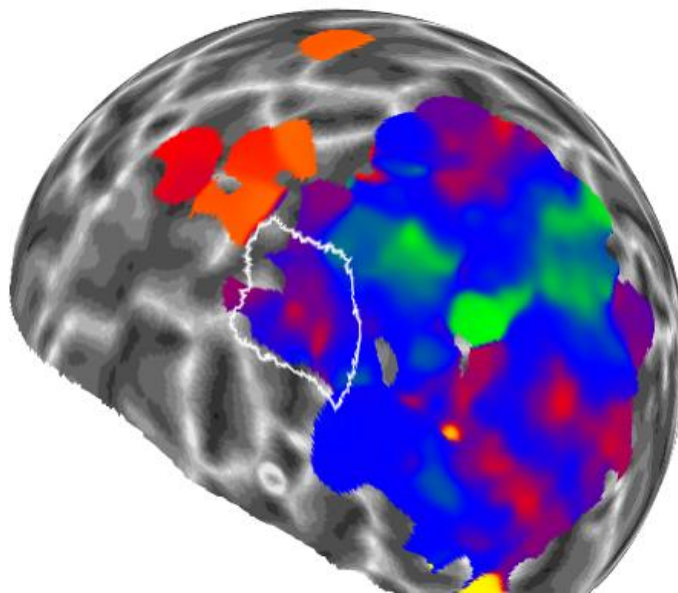
Critically, since this is the motion-sensitive MT+ complex, we should measure stronger responses to moving than static stimuli here. This is how original studies identified this region. They would therefore have been agnostic about any retinotopic structure that can distinguish sub-regions.

Below, I show the activation maps for a GLM contrast comparing alternating blocks of radially moving dots with blocks of static dots – a standard MT+ localiser scan. You can clearly make out a large activation all over the early areas V1-V3 and V3A. But there is also a clearly distinct cluster in medial temporal cortex. (Note that these maps are based on data that had been smoothed in the volume prior to the GLM analysis, hence the different granularity than in the pRF maps):



I drew a rough region of interest around this medial temporal activation. There is a little cluster in this particular map between that and the large one on the right. This is inside the LO complex and could correspond to parts of LO1 (which is probably what had been called KO, although that may also overlap with V3B). Anyway, we will not include this small patch but just the larger one. I now overlay this delineation on the retinotopic map:





The overlap is striking. This motion sensitive cluster quite clearly corresponds to the location of TO1 and TO2. Of course, things may not always work out this way and there are several factors that could screw up this relationship (such as bad coregistration or simply bad maps). But as far as I managed to see these TO areas and had a localiser at hand, the correspondence was usually very close. So in theory, if you get good pRF maps in these regions you should be able to map the MT+ complex without a motion localiser. However, a localiser may nevertheless be useful because it is a lot easier to identify when the retinotopic maps are poor.

Motion sensitivity may also be of interest elsewhere by the way. One thing I noticed is that V4 shows little response to the motion localiser, but the adjacent VO regions may do.

## Reading List

Here is a list of publications that you should look at for getting a better understanding of how these maps are organised. They also show you how large the individual differences can be and how there still remains disagreement about some of the regions:

### **Seminal study explaining visual area borders and field sign reversals**

Sereno, M.I., Dale, A.M., Reppas, J.B., Kwong, K.K., Belliveau, J.W., Brady, T.J., Rosen, B.R., Tootell, R.B., 1995. Borders of multiple visual areas in humans revealed by functional magnetic resonance imaging. *Science* 268, 889–893.

### **Good general overview over the visual field maps**

Wandell, B.A., Dumoulin, S.O., Brewer, A.A., 2007. Visual field maps in human cortex. *Neuron* 56, 366–383. <https://doi.org/10.1016/j.neuron.2007.10.012>

### **About the foveal confluence of the early regions**

Schira, M.M., Tyler, C.W., Breakspear, M., Spehar, B., 2009. The foveal confluence in human visual cortex. *J. Neurosci. Off. J. Soc. Neurosci.* 29, 9050–9058. <https://doi.org/10.1523/JNEUROSCI.1760-09.2009>

### **About the clusters with separate foveal confluences**

Wandell, B.A., Brewer, A.A., Dougherty, R.F., 2005. Visual field map clusters in human cortex. *Philos. Trans. R. Soc. Lond. B. Biol. Sci.* 360, 693–707. <https://doi.org/10.1098/rstb.2005.1628>

\* Please note that in later versions we changed the default colour schemes from what is shown here.

**About V4 organisation and the VO regions**

Winawer, J., Witthoft, N., 2015. Human V4 and ventral occipital retinotopic maps. *Vis. Neurosci.* 32, E020. <https://doi.org/10.1017/S0952523815000176>

**About the TO regions and also the LO regions**

Amano, K., Wandell, B.A., Dumoulin, S.O., 2009. Visual field maps, population receptive field sizes, and visual field coverage in the human MT+ complex. *J. Neurophysiol.* 102, 2704–2718. <https://doi.org/10.1152/jn.00102.2009>

**Original study on the LO regions**

Larsson, J., Heeger, D.J., 2006. Two retinotopic visual areas in human lateral occipital cortex. *J. Neurosci. Off. J. Soc. Neurosci.* 26, 13128–13142. <https://doi.org/10.1523/JNEUROSCI.1657-06.2006>

**TMS study looking at specialisation of the LO regions**

Silson, E.H., McKeefry, D.J., Rodgers, J., Gouws, A.D., Hymers, M., Morland, A.B., 2013. Specialized and independent processing of orientation and shape in visual field maps LO1 and LO2. *Nat. Neurosci.* 16, 267–269. <https://doi.org/10.1038/nn.3327>