Eurographics Workshop on Visual Computing for Biology and Medicine (2017)
Teodora Chitiboi, and Felix Ritter (Editors)

The Cortical Explorer: A Web-based User-interface for the Exploration of the Human Cerebral Cortex

Sam Budd[1], Emma C. Robinson[2] and Bernhard Kainz[1]

[1] BioMedIA, Imperial College London, UK [2] King's College London, UK

Abstract

Due to the immense complexity of the brain's neural network, research into the brain is performed at a hierarchy of scales. One common holistic approach is to model the brain, at a coarse scale, as consisting of a relatively small number of functionally specialised regions, connected in a network, known as the connectome. Recent advances in this field have led to a state-of-the-art mapping of the cortical connectome. This model has the potential to vastly increase our understanding of the mechanisms underpin an array of complex brain functions, and through this better understand the effects of health ageing, neurodevelopment, and neurological disease. However, often only those with specialised training understand what these different regions of the cortex actually do. This acts as a significant barrier for effective development of neuroimaging tech as most methods researchers do not understand the constraints or limitations of the data they are working with. To tackle this problem, we present Cortical Explorer*, a web-based user-interface for the exploration of human connectome data. Our tool allows users to explore the functional parcellation of the brain in a web browser in 3D. This facilitates easy access to parcel meta-data through intuitive interaction with individual parcels of the brain.*

Introduction: Recent advances in neuro-imaging have made it feasible to examine human brain connectivity systematically and across the whole brain in large numbers of individual subjects. The Human Connectome Project (HCP) [Hum17] produced a large and accomplished set of high quality neural data, with the objective of studying human brain connectivity and its variability in healthy adults. [VEUA*12]. Glasser *et al.* have used this unique collection of data to extract the most accomplished parcellation of the brain to date, splitting the brain into 180 distinct cortical areas per hemisphere [GCR*16]. Parcellation is the process of subdividing the brain's cortical surface into anatomically or functionally distinct regions. A parcellation forms a map of functionally connected areas of the brain. In order to establish differences between subjects it is vital that neuroscientists have a framework through which to compare brains, this is achieved by defining the brain as being composed of distinct regions.

The largest challenge faced by neuroscientists is that brain data is incredibly complex and non-intuitive to work with. Specialised training is needed to understand what different regions of the cortex do, and how the data relates back to the brain at a cellular level. This creates a significant barrier for the development of neuroimaging technology as the limitations and constraints of the data are rarely understand by those without special training. The neuroscience community traditionally works with volumetric data and there is a steep learning curve associated with processing and visualising surface based data. In contrast to this, the HCP promotes the surface modelling approach (representing data on the brains surface rather than volumetrically) as this has many advantages, as described in the HCP's pipeline paper [GSW*13].

1. Our Approach

A technical overview over our approach, which we call the *Cortical Explorer*, is shown in Figure 1. The following sections outline the components of the Cortical Explorer in detail.

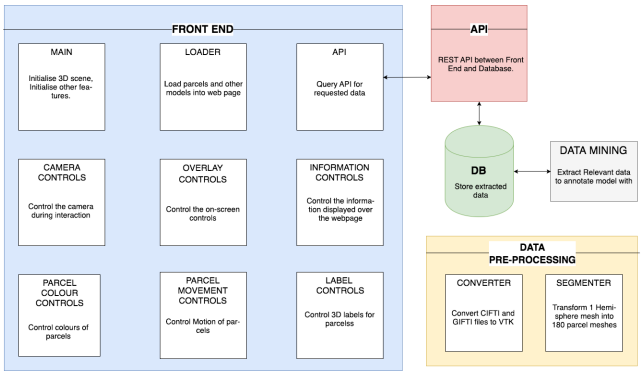


Figure 1: Overview over the proposed system

Framework: Our system is based on Node.js[†], which is a platform built on Chrome's JavaScript runtime. Node.js uses an event-driven, non-blocking I/O model that makes it perfectly suited for our main requirements: lightweight, efficient and suitable for data-intensive real-time applications that run on a large selection of de-

[†] https://nodejs.org/en/

vices. This framework was chosen for both the 'Front-End' and the 'API' as shown in Figure 1.

On top of this we use Ractive.js, which is a template-driven framework that aims to streamline building responsive, interactive user interfaces through the use of two-way binding[‡]. We use this framework for 'Overlay Controls' and the 'Front End' as shown in Figure 1.

Parcel information is stored in a document database, specifically MongoDB[§]. MongoDB allows for simple creation of document databases that can store data with a flexible schema. This is important as the data used will likely evolve throughout development. Node.js' package manager *npm* offers modules to simplify the interface between the application code and the MongdoDB database. The supporting meta-data for the parcels ('Data Mining' in Figure 1) comes in many different forms which means that the flexibility of MongoDB make it a good choice to store parcel meta-data in ('DB' in Figure 1).

3D data is managed and rendered using ThreeJS. ThreeJS is a JavasScript API for rendering interactive 2D and 3D graphics inside a HTML canvas element through a WebGL renderer. It is supported by all major browsers and works well on mobile and tablet devices.

Data conversion: HCP data is stored in two main formats in the BALSA database:

1. Surface GIFTI files (.surf.gii) - These files contain the vertices and face information needed for the 3D model.
2. Overlay CIFTI files (.dscalar.nii, .dlabel.nii) - These files contain per-vertex data such as myelin maps (per vertex scalar data) or parcel data (per vertex parcel index, dictionary of parcel index to rgba colour).

Contained inside a GIFTI surface file and CIFTI metric file are all the data necessary to produce a VTK model file. As CIFTI is still a relatively new standard with limited support from existing brain image data manipulation solutions, the Connectome Workbench's command line interface is used to convert the CIFTI files into more common GIFTI files before processing further.

This data needs then to be converted to VTK data, which is readable for ThreeJS. Converting GIFTI files into VTK requres three core steps as it is shown in Figure 2.

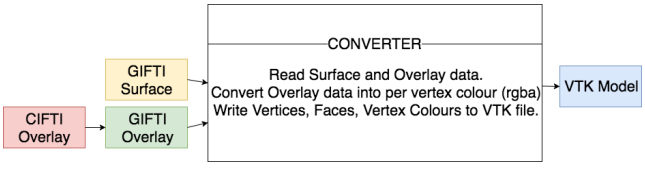
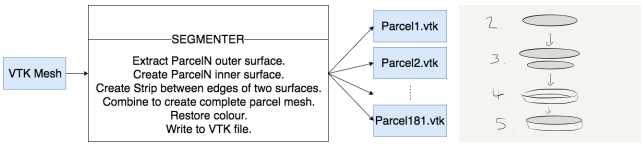


Figure 2: Overview over the data conversion process.

Parcel extraction: The whole brain mesh needs to be split into individual mesh objects to be able to interact with them individually. The Python VTK package was used to create 181 individual parcel meshes from the previously generated whole brain VTK mesh as shown in Figure 3(a). We designed an algorithm to split the parcels seamlessly into sub-meshes. Figure 3(b) gives a schematic overview over this process and algorithm 1 describes this process in detail.



(a) (b)

Figure 3: Overview over the parcel mesh generation step (a) ('Segmenter' from 'Data Pre-processing' in Figure 1). Schematic illustration of the steps in Listing 1 (b).

Algorithm 1 Parcel mesh generation

```
1:  procedure GENERATEPARCELS
2:      mesh ← load whole brain vtk mesh
3:      remove all vertices and faces from mesh that are not part of
    the target parcel.                                  ▷ Figure 3(b), 2
4:      parcel_outer ← mesh
5:      parcel_inner ← mesh
6:      translate parcel_inner towards mesh centroid
7:      shrink parcel_inner towards parcel_inner centroid
8:                          ▷ to create a tapered edge, Figure 3(b), 3
9:      parcel_strip = joinEdges(parcel_outer, parcel_inner);
10:                                                 ▷ Figure 3(b), 4
11:     parcel_mesh = join(parcel_outer, parcel_inner, parcel_strip);
12:                                                 ▷ Figure 3(b), 5
13:     store parcel_mesh as individual VTK file.
```

1.1. Front-end

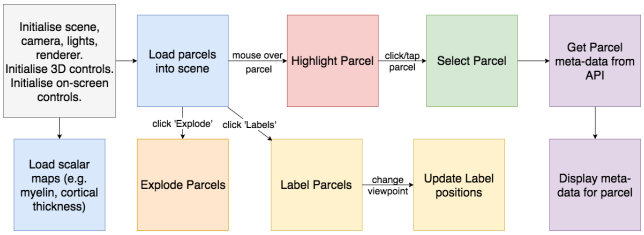


Figure 4: User flow through the Cortical Explorer 'Front End' (Starting in top left).

An overview over the browser-based front end is shown in Figure 4. ThreeJS is used to initialise the rendering loop and embed a 3D scene within the web-page. We use ThreeJS's 'Orbit Controls' to initialise interaction with the scene. 'Orbit Controls' supports zooming in/out, rotating camera around 'target' (The centre of the parcels in this case), and panning the scene. Sensible limits

‡ https://ractive.js.org/
§ https://www.mongodb.com/

are placed on these controls (e.g. users cannot zoom so far out that they cannot see the brain). Ractive is used to initialise on-screen controls that allow users to show/hide hemispheres of the brain by setting each of parcels in that hemisphere to invisible. For example, buttons to view the brain from common viewpoints (e.g. top, left, front), these update the camera position to the respective point and direction within the scene to view the brain from the chosen face.

When navigating the brain it must be clear to users when they have the option to interact with the parcels. When a user 'hovers' their mouse over a parcel, both change appearance to indicate the a parcel can be 'selected', this user flow can be seen in Figure 4. The 'hover' action is only applicable on desktop environments but concepts from 'highlighting' parcels are used to 'select' parcels as well.

By tracking the position of the mouse on the screen, ThreeJS's 'Raycaster' is used to 'cast' a ray into the scene and return a list of objects that this ray intersects, *i.e.*, picking of objects that are underneath the mouse in the scene. Intersection is calculated by checking whether the triangles that make up each object intersect with the ray. The first element of the returned list is the object under the mouse that is closest to the camera and the most visible object under the mouse. This object is 'highlighted'. To highlight the object, the object's 'Emissive' property is updated. The 'Emissive' property lets an object emit a coloured light, this defaults to a black light that has no effect on the appearance of the model.

To 'Select' a parcel, two approaches can be chosen: (1) Drag Parcel: When the user clicks/taps on a parcel, the parcel becomes selected and the user can manually drag the parcel along a line that minimises overlap with other parcels. The user drags upwards to move the parcel away from the brain and downwards to move the parcel towards its original position. When the user releases the mouse the parcel stays in that place. (2) Pop-out Parcel: When the user clicks/taps on a parcel, the parcel moves a fixed length along a line outwards of the brain, clicking/tapping again will either raise the parcel another length along the line or move the parcel back to its original position. Users are able to toggle between these two strategies.

1.2. Meta information

Similar to [GCR*16] we define a parcel's colour by a scheme that reflects broad classes of brain function - these are 'Auditory', 'Visual' and 'Sensory and Motor'. A colour legend is shown in the upper left corner of Figure 6.

Supplementary neuro-anatomical information used to delineate areas, as well as background information on their possible functions are available in neuroscientific literature. When selecting a parcel we display this information as an additional overlay. The current selection comprises of:

- *Parcel description* - short description of each parcel.
- *new area?* - indication of whether this region has been reported earlier than [GCR*16].
- *study* - hyper links to scientific papers that describe the selected parcel.
- *aka* - other common names for the selected parcel.
- *sections* - highlights the anatomical sections that the parcel is part of.
- *top connection* - highlights the top 5, 10 or 50 other parcels that are most strongly connected to the selected parcel.

The user can toggle between two different methods of displaying these collections of parcels: (1) Pop-out parcels, for which each parcel 'pops out' to an elevated position above the brain and (2) recolour parcels: All parcels that are not in the selected 'collection' are faded from view by updating their emissive property to a white colour.

1.2.1. Labelling Parcels

Without pre-existing knowledge of the parcellation, it is difficult for users to find specific parcels. To aid the user searching for specific parcels, we provide a labels toggle that turns labels of the parcels on or off . We use a Labels-on-a-pole metaphor: Labels appear at the end of a line attached to the relevant object, this makes labels more clearly visible and offers more flexibility in where the label and pole are placed.

For each parcel, a pole start point is calculated first by calculation of the centre of the bounding box of the parcel and then finding the closest vertex on the parcel to the centre point. This vertex position is used as the starting point for the line. The end point is calculated by: $p_{endpoint} = p_{startpoint} + \alpha * l_{line}$ and $l_{line} = p_{startpoint} - c_{centrepoint}$, where α is a heuristically defined constant and $c_{centrepoint}$ updates to either the centre point in-between the two brain hemispheres when both hemispheres are visible, or to the centre point of individual hemispheres when only that hemisphere is visible. A 2D canvas displaying the name of the parcel is created for each label, this is placed at the endpoint of the pole. The text label always faces the camera as a billboard.

A problem with this approach is that labels themselves may occlude other parcels and/or other labels. We utilize a simple but fast version of smart labelling as proposed by Tatzgern et al. [TKGS14]. Each label is placed at the end of a pole extruding from the parcel, and moves along this pole in real-time to a position where it does not block the view of another label. This is achieved by calculating the screen position of each label: $p_{screenposition} = MVP * p_{worldposition}$, where MVP is the Mode- View-Projection matrix of the camera. Calculating the size of the label for each parcel allows calculation of whether these rectangles intersect, if a label occludes other labels it is incrementally moved along its pole until it meets satisfactory conditions.

2. Evaluation & Results

In this section we present early qualitative results from a small expert user study (N=5 Neuroscientists, Figure 5) and quantitative results in terms of data processing speed and real-time performance. Table 1 details the performance of each of the core front-end features on MacBook 7. The current front-end is shown in Figure 6.

2.1. Conclusions

The Cortical Explorer is web-based and device-independent. This greatly increases the accessibility of brain parcellation insights and

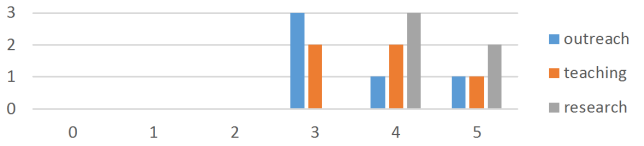


Figure 5: Qualitative expert survey (N=5 Neuroscientists). Responses to Question: How useful is the Cortical Explorer to the general public? (outreach) How useful is the Cortical Explorer as a teaching tool? (teaching) How useful is the Cortical Explorer as a research tool? (research) (x-axis = Likert-type scale: 0 = Not at all, 5 = Extremely useful; y-axis = votes)

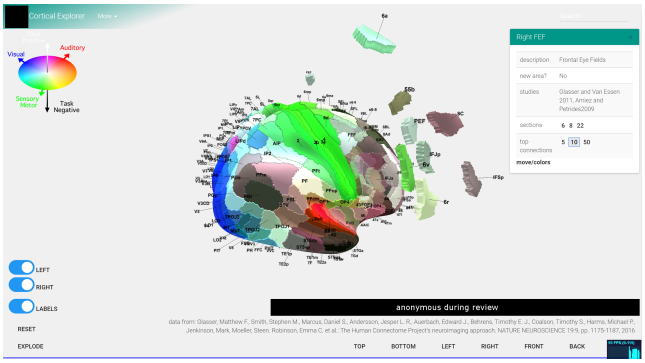


Figure 6: Current Cortical Explorer web front-end.

Table 1: Feature performance

Feature	FPS	Loading Time
Parcels	85	45s, 11s from cache
Highlighting Objects	75	negligible
Selecting Objects	65	negligible
Dragging Objects	75	negligible
View Section	75	negligible
View Collection	75	negligible
Explosion	35	negligible
Labels	35	5s, negligible once loaded
Hide Hemisphere	110	negligible
'More' Brains	120	40s, 7s from cache
Alternate Parcel surface	65	25s, negligible once loaded

supporting meta-data. Our approach is a novel solution to visualising surface based brain data on the web. It takes advantage of the flexibility of Node.js and the VTK model format. This paves the way for visualising surface based brain data that offers new ways of exploring brain parcellation data. With further work, the Cortical Explorer has the potential to become a widely-used tool for brain image analysis and visualisation. The core features of the Cortical Explorer are currently deployed on a publicly available virtual machine. *The link will be provided here after the double-blind review phase.*

Ou main target audiences are:

1. The General Public: To make the Cortical Explorer of more interest to the public I propose including general neuroscience information as well as limited scientific data that is more colloquial. Including explanations of how this data was collected, the different imaging modalities and how the cortex relates the rest of the brain would be beneficial. With enough relevant information present this would make the Cortical Explorer an effective teaching tool.
2. Neuroscientists: Several features that would be useful to neuroscientists using the types of data my project has discussed were proposed to me while gathering feedback. These fall into three categories:
   a. Analysis: providing back-end tools to perform analyses on user uploaded data.
   b. Presentation: allowing users to upload their own data for visualisation and analyses.
   c. Exploration: extending existing functionality to allow users to explore and annotate their own data.

The Cortical Explorer is Web-based and device-independent, this greatly increases the accessibility of brain parcellation and supporting meta-data. Allowing users to explore per-parcel data via interaction within a structured 3D scene is an objective improvement on exploring by reading a text file.

Overall the Cortical Explorer is a powerful tool for visualising surface based brain data that offers new ways of exploring this data. With further work the Cortical Explorer has the potential to become a widely-used tool for brain image analysis and visualisation.

References

[GCR*16] GLASSER M. F., COALSON T. S., ROBINSON E. C., HACKER C. D., HARWELL J., YACOUB E., UGURBIL K., ANDERSSON J., BECKMANN C. F., JENKINSON M., SMITH S. M., VAN ESSEN D. C.: A multi-modal parcellation of human cerebral cortex. *Nature Publishing Group 536* (2016). doi:10.1038/nature18933. 1, 3

[GSW*13] GLASSER M. F., SOTIROPOULOS S. N., WILSON J. A., COALSON T. S., FISCHL B., ANDERSSON J. L., XU J., JBABDI S., WEBSTER M., POLIMENI J. R., VAN ESSEN D. C., JENKINSON M.: The minimal preprocessing pipelines for the Human Connectome Project. *NeuroImage* (2013). doi:10.1016/j.neuroimage.2013.04.127. 1

[Hum17] Human Connectome Project Website, 06 2017. http://www.humanconnectomeproject.org/. 1

[TKGS14] TATZGERN M., KALKOFEN D., GRASSET R., SCHMALSTIEG D.: Hedgehog labeling: View management techniques for external labels in 3D space. In *2014 IEEE Virtual Reality (VR)* (3 2014), IEEE, pp. 27–32. http://ieeexplore.ieee.org/document/6802046/. doi:10.1109/VR.2014.6802046. 3

[VEUA*12] VAN ESSEN D. C., UGURBIL K., AUERBACH E., BARCH D., BEHRENS T. E. J., BUCHOLZ R., CHANG A., CHEN L., CORBETTA M., CURTISS S. W., DELLA PENNA S., FEINBERG D., GLASSER M. F., HAREL N., HEATH A. C., LARSON-PRIOR L., MARCUS D., MICHALAREAS G., MOELLER S., OOSTENVELD R., PETERSEN S. E., PRIOR F., SCHLAGGAR B. L., SMITH S. M., SNYDER A. Z., XU J., YACOUB E.: The Human Connectome Project: A data acquisition perspective, 2012. doi:10.1016/j.neuroimage.2012.02.018. 1