Table of Contents

Supplementary material

Web page http://www.3dbar.org/wiki/barSupplement contains the most recent versions of supplementary materials for the article:

Common Atlas Format and 3D Brain Atlas Reconstructor: Infrastructure for Constructing 3D Brain Atlases

Piotr Majka, Ewa Kublik, Grzegorz Furga, Daniel K. Wójcik (Neuroinformatics 2011, http://dx.doi.org/10.1007/s12021-011-9138-6)

ABSTRACT

One of the challenges of modern neuroscience is integrating voluminous data of diferent modalities derived from a variety of specimens. This task requires a common spatial framework that can be provided by brain atlases. The first atlases were limited to two-dimentional presentation of structural data. Recently, attempts at creating 3D atlases have been made to offer navigation within non-standard anatomical planes and improve capability of localization of different types of data within the brain volume.

The 3D atlases available so far have been created using frameworks which make it difficult for other researchers to replicate the results. To facilitate reproducible research and data sharing in the field we propose an SVG-based Common Atlas Format (CAF) to store 2D atlas delineations or other compatible data and 3D Brain Atlas Reconstructor (3dBAR), software dedicated to automated reconstruction of three-dimensional brain structures from 2D atlas data. The basic functionality is provided by 1) a set of parsers which translate various atlases from a number of formats into the CAF, and 2) a module generating 3D models from CAF datasets.

The whole reconstruction process is reproducible and can easily be configured, tracked and reviewed, which facilitates fixing errors. Manual corrections can be made when automatic reconstruction is not sufficient. The software was designed to simplify interoperability with other neuroinformatics tools by using open file formats. The content can easily be exchanged at any stage of data processing. The framework allows for the addition of new public or proprietary content.

Following supplementary materials are available:

1. Description of the graphical user interface for reconstruction module;
2. Command-line reconstruction interface manual;
3. Description of vector data processing - typical problems and their solutions;
4. Detailed description of the gap-filling algorithm;
5. Creation of new structures.

Description of GUI of the 3D model building tool

To facilitate reconstruction process a simple graphical user interface was prepared. Below, the 3dBAR reconstruction GUI with an exemplary CAF dataset loaded (Johnson et. al., 2010.)

Selecting structures

The left panel contains *ontology tree* loaded from CAF index file. One can browse through structures, select them for reconstruction or load already reconstructed, cached models. The right panel contains two tabs. *Structure selection tab* displays detailed information about the currently selected structure for reconstruction. This includes abbreviated and full name of the structure, span of slides on which it is defined as well as color assigned to this structure. Here one can also set the resolution, the depth of decomposition of the structure into substructures according to ontology tree, and mirroring of brain hemisphere may be enabled if needed.

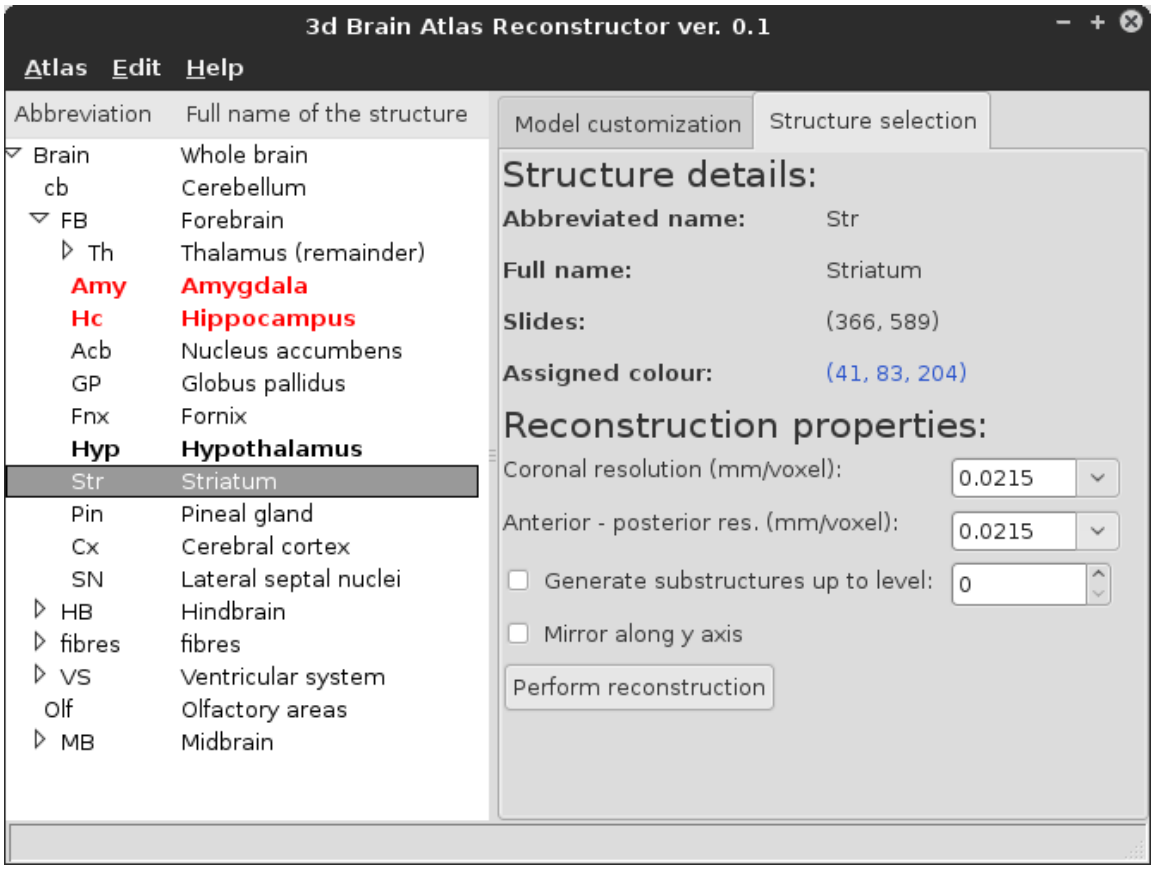


Figure 1: The *structure selection tab*. The *Striatum* selected for reconstruction (gray highlight); *Amygdala* and *Hippocampus* loaded as context models (red font). The reconstruction of `Hypothalamus` is cached and available for loading (bold font).

Viewing and customizing the reconstructions

Reconstructed models can be viewed in the *Model customization tab* (Fig. 2) which also provides access to parameters of VTK pipeline. The preview is displayed in the form of triangular mesh although the actual model is still stored in volumetric form. The model can be rotated by pressing and holding left mouse button,

zoomed in or out by pressing and holding the right button. Pressing the middle button allows one to pan the model over the scene. One can switch between wireframe and surface representations of the model by pressing `w` or `s` keys.

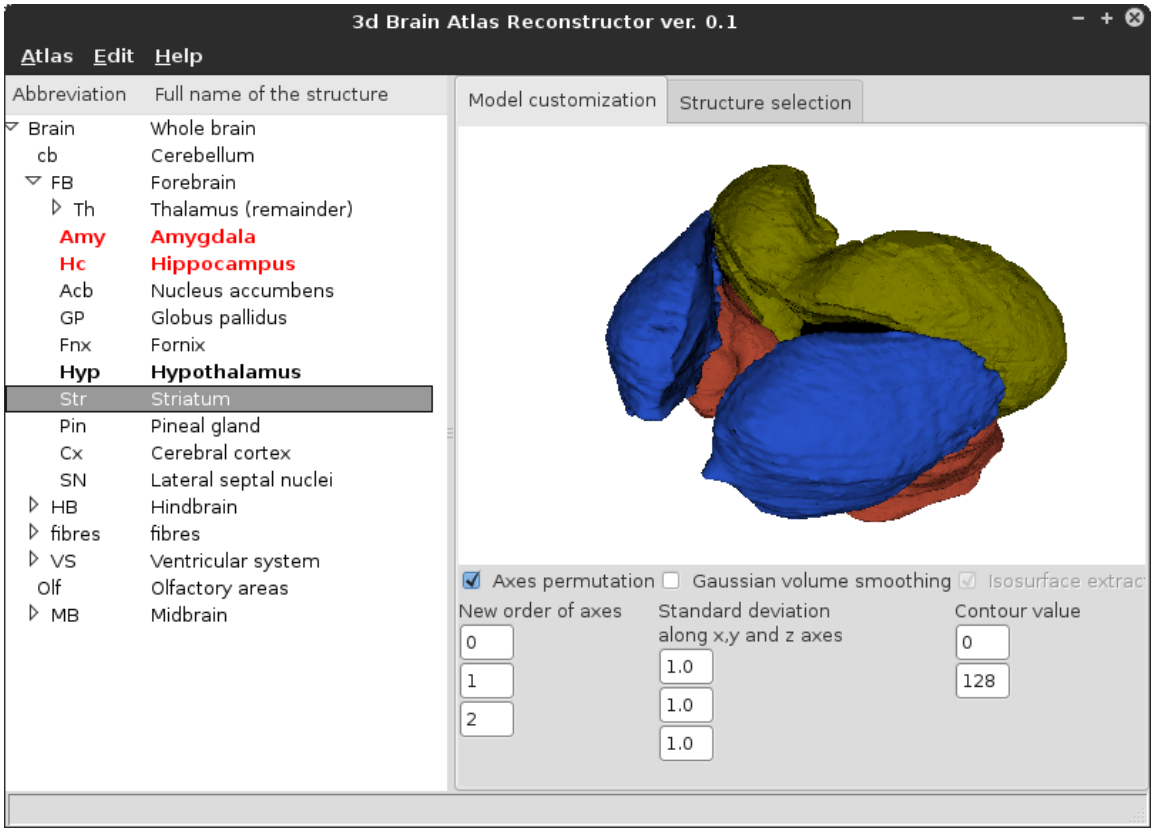


Figure 2: The *model customization tab*. Reconstructions of *Striatum*, *Amygdala* and *Hippocampus* are presented.

After changing VTK pipeline parameters, mesh may be recalculated by pressing *Refresh* button and cached by pressing *Save model*. Models are cached in form of polygonal meshes (vtkPolyData). *Clear scene* button removes all models from preview window. When a model is cached, the corresponding element of the ontology tree is displayed with bold font. Cached models can be loaded in the preview window as *context models* alongside the model being presently reconstructed, however they cannot be modified. When a cached model is loaded into context, the corresponding element in ontology tree is displayed in red.

Generating multiple reconstructions

Multiple reconstructions can be performed one by one as described above. However, this is convenient only when few models are required. To reconstruct multiple structures one can choose a parent structure containing desired substructures and reconstruct them down to a desired level of detail. All substructures are generated using the same set of reconstruction parameters provided in `Structure selection tab` and the same VTK pipeline. Models are generated one by one and automatically cached. When all the substructures are reconstructed they are loaded into preview window as context models.

Exporting the models

By default models are cached in the directory where CAF dataset is located. One may choose another export directory by selecting *Set cache directory* from *Edit* menu. When cache directory is changed, all currently loaded structures are removed from preview and ontology tree updates information about availability of cached models from the new directory.

3d Brain Atlas Reconstructor GUI offers three ways of exporting three dimensional models. Single reconstructions may be cached as polygonal data (vtkPolyData) or exported as volumetric datasets in the native VTK (vtkImageData) or (Neuroimaging Informatics Technology Initiative, NIfTI) formats. Complex scenes containing numerous context models may be exported into (Virtual Reality Modelling Language, VRML) or X3D format.

Command-line reconstruction interface

Apart from GUI, the 3D Brain Atlas Reconstructor comes with command line interface (or shorter: CLI) allowing the user to perform batch reconstructions without configured graphics environment. After invoking the batch interface, following information will appear:

```
$./batchinterface.sh

Usage: ./batchinterface.sh [options] <CAF index> [<structure 1> [<structure 2> ...]]
3d Brain Atlas Reconstructor ver. 0.1 Batch reconstruction interface

Options:
-h, --help            show this help message and exit
-g GENERATESUBSTRUCTURES, --generateSubstructures=GENERATESUBSTRUCTURES
                      maximum level of substructures (in the structure tree)
                      to be generated; default: 0
-d VOXELDIMENSIONS, --voxelDimensions=VOXELDIMENSIONS
                      voxel size (in units defined in given dataset) in slide plane
                      and in plane perpendicular to the slides, consecutively.
-e EXPORTDIR, --exportDir=EXPORTDIR
                      the path to a directory for reconstructions
-p PIPELINE, --usePipeline=PIPELINE
                      the path to a custom pipeline definition
-v CAMERA, --useViewport=CAMERA
                      the direction vector from the center of the scene to
                      the camera position
--exportToWindow, --show
                      the reconstruction is displayed to the user
--composite           perform a reconstruction of the structure as a scene
                      composed of the reconstructions of the basic
                      substructures in the hierarchy tree (up to the maximum
                      given tree depth, see -g switch)
--includeBrainOutline
                      Includes additional translucent brain outline to the
                      reconstructions. Applies only when exporting to VRML,
                      X3D, screenshot or thumbnail.

Output Format Options:
  --exportToX3d       exports as X3D scene
  --exportToVRML      exports as VRML scene
  --exportToVTKPolydata
                      exports as vtkPolyData
  --exportToVolume    exports as vtkStructuredPoints
  --exportToNifti     exports as NIfTI file
  --exportToNumpy     exports as NumPy array
  --exportScreenshot  saves screenshot as an PNG image
  --exportThumbnail   saves scaled screenshot as an PNG image
```

The `<CAF index>` is a location of the index file of the given CAF dataset while the `[<structure 1> [<structure 2> ...]]` is space-separated list of all structures to reconstruction contained in the CAF dataset.

The simplest usage of the CLI is to generate reconstruction of a single structure from provided CAF dataset using default settings. Assume that our CAF dataset is located in the `~/atlases/mouse/caf/` directory and we would like to reconstruct the *Thalamus* and save a nifti volume with the reconstruction:

```
$./batchinterface.sh ~/atlases/mouse/caf/index.xml Thalamus --exportToNiftii
```

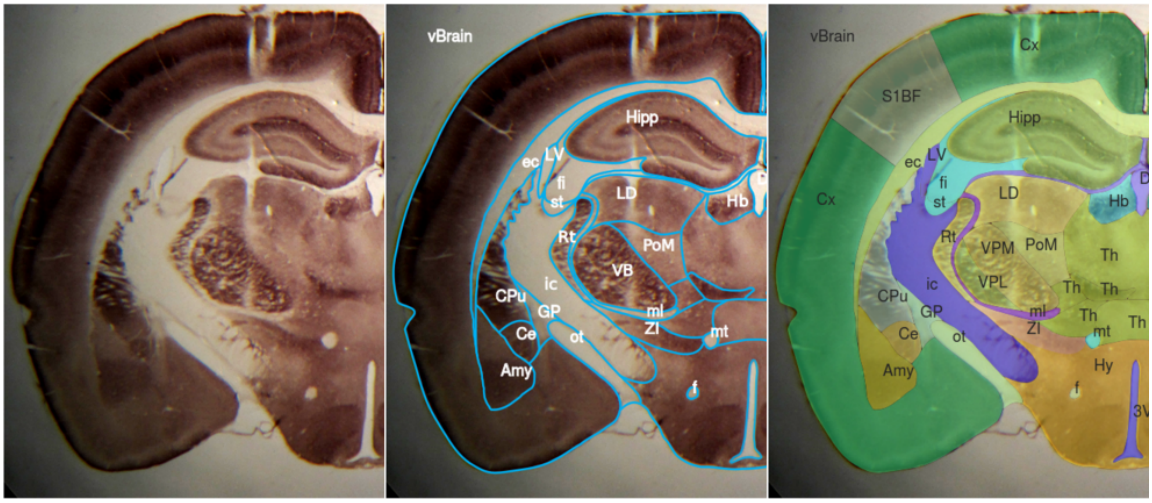
By default, the reconstructions generated using a predefined pipeline and stored in directory parallel to the CAF dataset directory (in this example it would be `~/atlases/mouse/reconstructions/`). However, this behavior can be customized using various switches.

Description of vector data processing - typical problems and their solutions

Contour slide

In our work-flow we introduced a stage of *contour slides* - vector graphic in SVG format (for detailed description see Majka et al 2011) that simplifies necessary corrections, manual preparation of structures' outlines from histology results and further improvements of slide details.

FIGURE 1.
Left: Coronal slice of a rat brain around 3 mm behind Bregma stained for Cytochrome Oxidase. Histological verification of the electrodes' location after chronic local field potentials (LFP) recording from the thalamic somatosensory nuclei. **Middle**: Initial stage of manual preparation of a *countour slide*. Main structures outlines drawn over the histology slice (line thickness increased for better visibility) and structure labels placed over well recognized regions. **Right**: Histological slice visible through translucent final CAF slide.



Typical problems and their solutions

Most commonly available 2D atlases were not designed to be used as sources for 3D models generation. Precision of contour definitions and label placement is sufficient for standard use, but not always for the automatic reconstruction process. For example, in some published atlases one may find a number of deliberate simplifications e.g. labels are omitted because names of some areas are considered obvious or labels are literally too large to put them inside the represented area. Such omissions have to be fixed in order to get a correct 3D reconstruction, indeed for any systematic computer processing.

Manually prepared *contour slides* - including our example presented here - also can contain errors requiring corrections.

Non-interactive error correction

The vector parser tries to automatically deal with troublesome situations. However, if it is unable to correct a problem automatically, the program will notify the user and save all the important information about the case

as a thumbnail image including name of the problematic label and its location. These data help the user to solve the problem manually by removing ambiguities from contour slides. Below we discuss and illustrate common inconveniences and our strategies for fixing them.

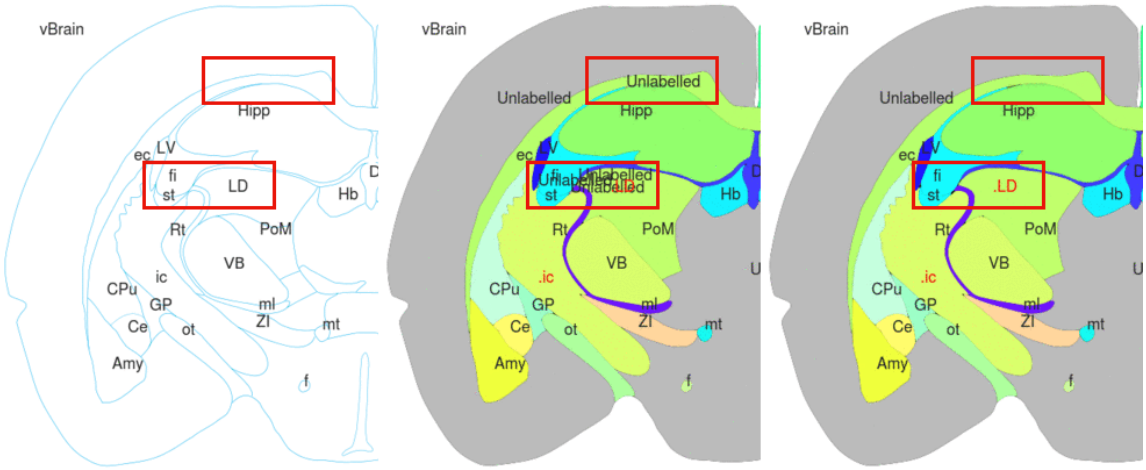
Labels

Unlabelled areas

A problem we encountered during tracing was the lack of labels attributed to some areas. This can happen when "obvious" names are omitted in commercial atlases; when user does not recognize the area in histological slice (see example in Creation of new structures), but also when additional area is erroneously defined by accidental line crossing (see figure 2 below).

3dBAR vector parser automatically detects such unlabeled regions by comparing the sum of traced paths with the total area of the *Brain* structure. The parser locates all contiguous unlabelled regions consisting of more than a given, configurable number of pixels and traces them. The resulting structures are called *Unlabelled* and may be indexed and reconstructed similarly to the regular structures.

FIGURE 2.
Left: Red blocks mark regions where irregular neighboring lines touch each other creating small closed areas. **Middle**: Small areas were detected by the parser, traced and marked as Unlabelled. **Right**: Adjusting the threshold number of pixels that should be traced by parser excluded incorrectly recognized tiny structures.



Duplicated labels within a single contour

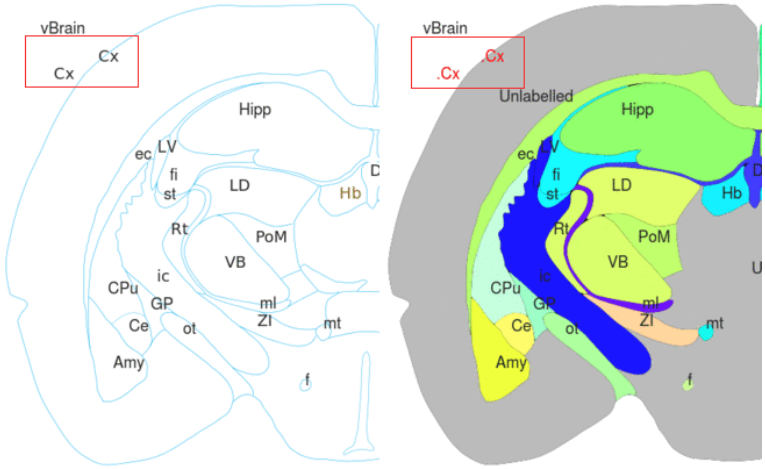
Another common difficulty is an absence of borders drawn within an outline containing multiple labels. This can be a consequence of improper drawing but often simply follows from lack of sharp transitions between structures. When the parser detects two or more overlapping regions that were created based on different labels such labels are changed to spot labels in the CAF slide and a notification is logged. Neither path is removed as there is no way to determine which one should be discarded and which should be kept. After tracing, the user is notified about ambiguous delineation and should review the slide in question and correct it. In the CAF slide on the right of figure 2 above we can see two labels (ic, Rt) located within one region. One of them was changed by parser to spot label (.ic). This particular example resulted from improper drawing and a gap existing in the border between internal capsule (ic) and reticular nucleus of the thalamus (Rt). The

parser treated areas connected by the gap as one structure (see *leaking of the structure*, below).

Incorrect label placement

Error correction procedure also looks for labels displaced outside the brain outline or labels anchored directly above contours.
The latter may happen when a structure is defined as a thin strip between other structures. Since tracing contours themselves is not allowed, such incidents are recorded and a thumbnail image is generated allowing user to examine and correct the error.
Tracing of a region corresponding to the incorrectly placed label is skipped but in general tracing is not interrupted as the whole procedure works non-interactively.

FIGURE 3.

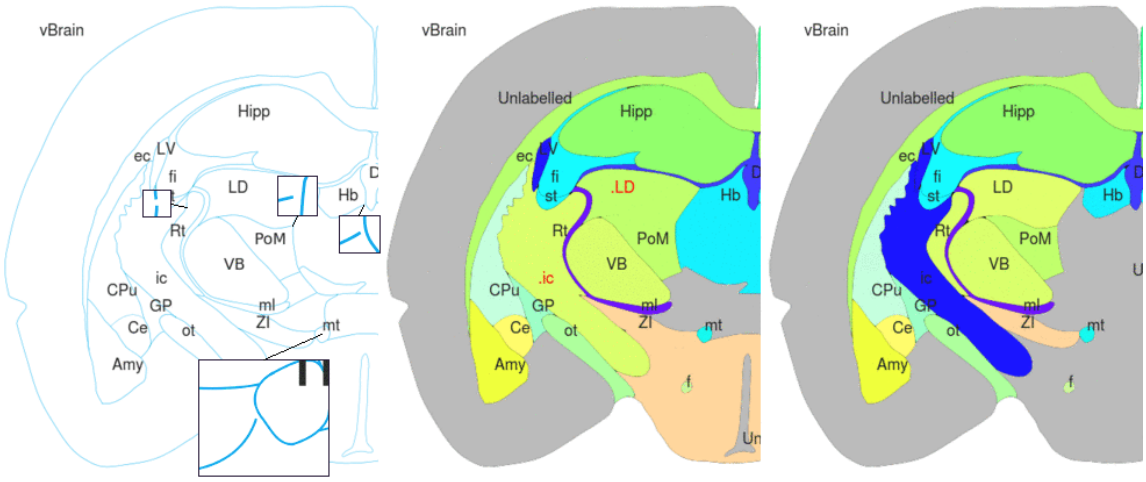


Leaking structures and gap filling algorithm

The most common drawing defect is gaps in structure outlines. It frequently happens when two contour lines, which should touch are drawn so that they leave a little space in between. Such an arrangement may not be visible in a printed atlas but it greatly disturbs the tracing process where every pixel may influence the results. In this case, the structure being traced overtakes the space of its neighbour through the broken contour which we call *leaking of the structure* (see inserts in figure 4).

It is handled by a heuristic gap filling algorithm (detailed description). The main idea behind this algorithm is to expand the contours by applying a dilation filter (Gonzalez2008) until the boundary closes. If a very accurate reconstruction is required, the best policy is to prepare a precise contour slide so there is no need to apply gap filling. For the atlases we tested, the gap filling algorithm performed well, the reconstructed structures were adequate and the gains in time were tremendous as compared with manual cleaning.

FIGURE 4.
Left: Contour slide containing small holes in contours. **Middle**: CAF slide without *gap filling* enabled. ic,Rt and PoM, Ld pairs of structures are blended together. ZI overtakes the space of hypothalamus while Hb - space of thalamus. **Right**: Analogous CAF slide with gap filling enabled. Structures are correctly divided.


vBrain
Hipp
ec
LV
fi
LD
Hb
Rt
PoM
VB
ic
CPu
GP
ml
Ce
ot
ZI
mt
Amy
vBrain
Unlabelled
Hipp
ec
LV
fi
st
.LD
Hb
Rt
PoM
VB
.ic
CPu
GP
ml
Ce
ot
ZI
mt
Amy
vBrain
Unlabelled
Hipp
ec
LV
fi
st
LD
Hb
Rt
PoM
VB
CPu
GP
ml
Ce
ot
ZI
mt
Amy

Detailed description of gap-filling algorithm

The most common defect encountered in digital (e.g. PDF) versions of published atlases are gaps in structures' outlines. It commonly happens when two contour lines which should touch are drawn so that they leave a little space in between. Such an arrangement may not be visible in a printed atlas but it greatly disturbs tracing process where every pixel may influence the results. In this case the structure being traced overtakes the space of its neighbour through the broken contour which we call leaking of the structure (see figure below).

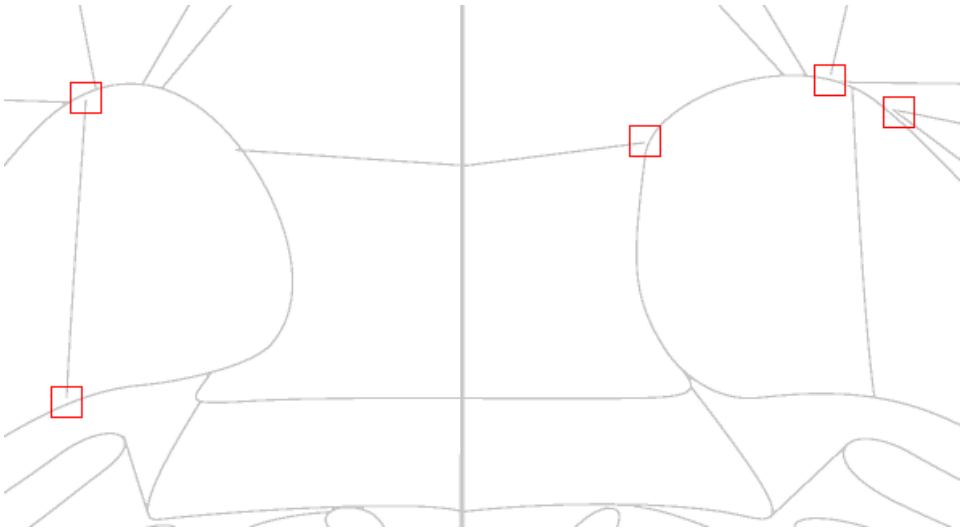


Figure 1: Examples of gaps in structures contours.

Fragment of rasterized contour slide based on Paxinos and Franklin *The Mouse Brain In Stereotaxic Coordinates, third edition*.

Leaking structures are handled in 3dBAR by a *gap filling* algorithm, an extension of the basic tracing procedure. The main idea behind this algorithm is to expand the contours by applying dilation filter until the boundary closes.

The algorithm requires a single input parameter - `MaxGrowLevel` - describing the maximum size of the gaps that may be closed during tracing of a particular area, and works as follows: First, the initial bitmap is duplicated. Dilation filter is applied to the copy and the result is also cached. This procedure is repeated `MaxGrowLevel` times. Each application of the dilation filter causes the boundaries of the structures to grow approximately 1 pixel in diameter. As a result, `MaxGrowLevel+1` bitmaps are cached with the first being the rasterized version of the original contour slide (see Figure 2 below).

Figure 2: Consecutive application of dilation filter to a rasterized contour slide.

Structure areas on each cached bitmap are flood-filled and then dilated the same number of times as the boundaries were grown to ensure that the area of the resulting path stays unbiased. If it remains stable or slightly lowers (less than 2% in our implementation) at consecutive stages of the algorithm it means that most probably the boundaries of the structure in question do not have any gaps (see Fig. 3).

Figure 3: Structure without gaps does not express significant changes of its area across different stages of the algorithm (see table below).

growlevel	area of the structure in pixels
0	51124
1	51124
2	51124
3	51124
4	51124
5	51112

If the reduction of flooded area at certain stage of filter application is greater than a defined threshold (15-20% in our implementation), it means that a gap in the boundary must have been closed (see Figure 4 below). After finding the optimal number of border expansions (denoted by `growlevel`) the corresponding bitmap is passed to PoTrace.

Figure 4: Structure with gap(s) in it's contour expresses significant changes of its area. We can see sudden drop of the number of pixels at certain stage (here at `growlevel=2`) of the algorithm application when the gap is closed (see table below).

growlevel	area of the structure in pixels
0	196243
1	**196232**
2	**84384**
3	84331
4	84331
5	84309

The gap filling algorithm is applied individually for each seed label thus different regions may be processed with different values of `growlevel`. Information about which `growlevel` was used to trace a particular path is stored in its attributes.

Preconfiguration

The gap filling algorithm may be preconfigured during the preparation of a contour slide. The preconfiguration is carried out by assigning `bar:growlevel` attribute to the label used as the seed for tracing. The value of this attribute should be an integer between -1 and `MaxGrowLevel`. Assigning `-1` means that `growlevel` is undefined and has to be selected automatically, `0` means that gap filling should not be used for this particular label. Other values denote the number of dilation filter applications.

Algorithm efficacy and limitations

The table below summarizes results of the gap filling algorithm applied to 47 paths from 13th coronal section (2.1mm ant. from bregma) from Paxinos and Franklin *The Mouse Brain In Stereotaxic Coordinates, third edition*.

optimal grow level	number of paths
0	28
1	9
2	10

The performance of the algorithm depends on the initial size of the structure and on the bitmap resolution. Intensive use of this algorithm may distort or even erase some (particularly narrow) structures. There is no

specific mechanism implemented against such behavior. If a very accurate reconstruction is required, the best policy is to carefully prepare contour slides adjusting stroke width and slice resterization resolution so there is no need to apply gap filling.

Definition of new structures

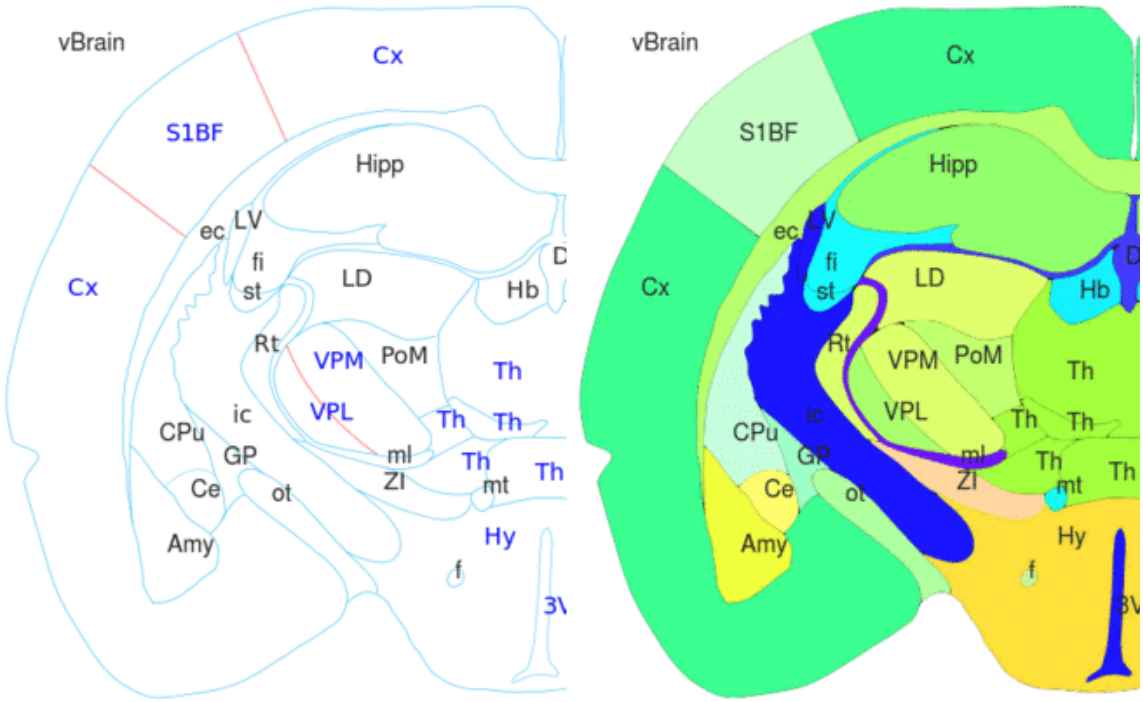
Substructures singled out by simple line drawing

Splitting given structure into smaller substructures comes down to drawing dividing lines or new contour and placing corresponding labels. Redefining the shape is equivalent to editing its contour.

Unrecognized areas

Unrecognized areas in a slide can be attributed to the structures from the upper hierarchy level.

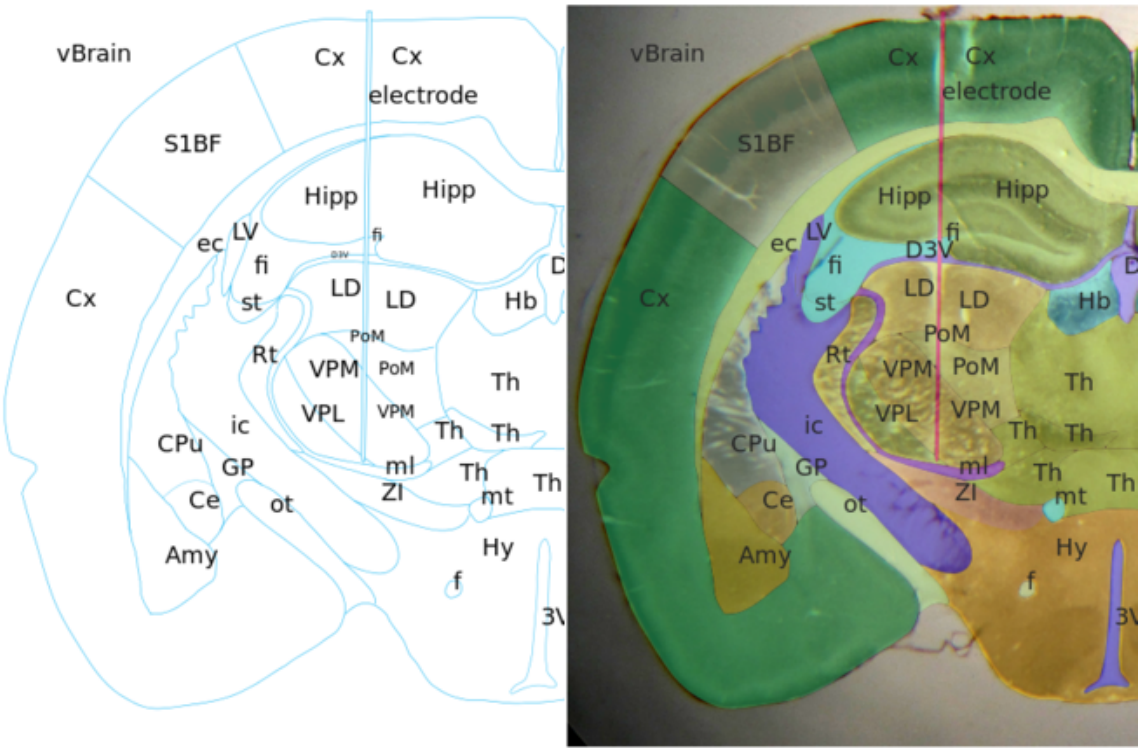
FIGURE 5.
Left: New lines and appropriate labels (blinking) were added to the *contour slide* 1) to show primary somatosensory representation of big facial whiskers ("barrel field", S1BF) in the cerebral cortex (Cx); and 2) to divide ventrobasal complex (VB) into ventral posteromedial (VPM) and ventral posterolateral (VPL) nuclei. Unrecognized thalamic nuclei, which were not labeled with their specific names, are labeled with "Th" so they all will be included in the final reconstruction of the thalamus. **Right**: Updated CAF slide.



Additional elements apart anatomical data included in CAF slide

In our example we include in the slide an outline of the electrode. Spaces occupied by lesions or injected dyes can be also added to atlas elements.

FIGURE 6.



List of used abbreviations

- 3V - 3rd ventricle
- Amy - nuclei of amygdala
- Ce - central amygdaloid nucleus
- CPu - caudate putamen
- Cx - cerebral cortex
- D3V - dorsal 3rd ventricle
- ec - external capsule
- f - fornix
- fi - fimbria of the hippocampus
- GP - globus pallidus
- Hb - habenula
- Hipp - hippocampal formation
- ic - internal capsule
- LD - laterodorsal thalamic nucleus
- LV - lateral ventricle
- ml - medial lemniscus
- mt - mammillothalamic tract
- ot - optical tract
- PoM - posteromedial thalamic nucleus
- Rt - reticular thalamic nucleus
- S1BF - primary somatosensory cortex, barrel field
- st - stria terminalis
- VB - ventrobasal complex
- vBrain - compliment of the whole brain outline
- ZI - zona incerta