4 Graphical User Interface Labelling Tool

In this chapter the main features of the labelling tool are presented. Furthermore, chapter 4.2 and 4.3 go into more detail on how the features were implemented and work in the background, and how to best use the GUI.

4.1 Features

4.1.1 Menus

In figure 4.1 the start window is shown. The annotation background is set to the extracted cellmap of the PCA-ICA method (for now) and the menu for the different features are visible in the top. When hovering over the different menus possible actions are shown (figure 4.2). To minimize the needed labelling time, almost all actions can also be triggered by a keyboard shortcut which are explained in chapter 4.3. Some actions such as the zoom function can only be triggered by keyboard shortcuts.

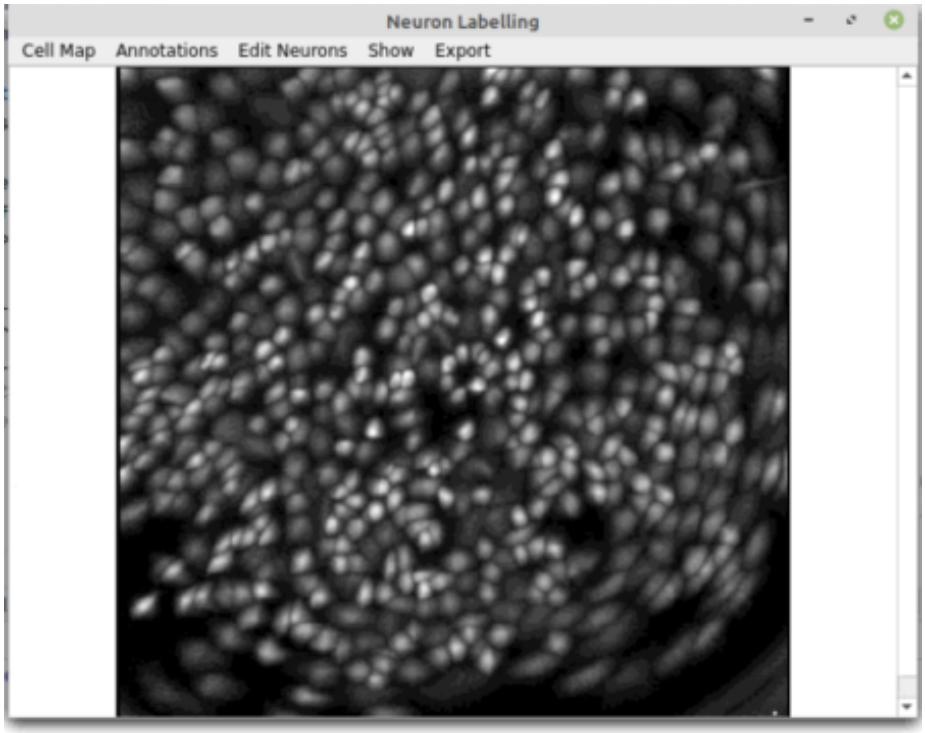

Figure 4.1: Main Window of the Labelling Tool

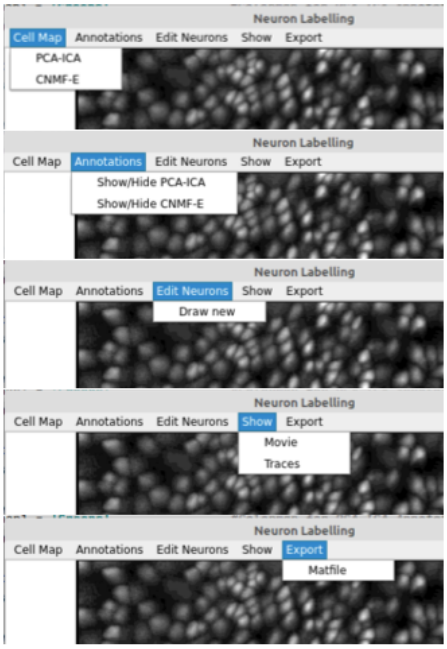

Figure 4.2: Dropdown Menus of the Main Window

4.1.2 Annotations

Once the main window is loaded the different extracted ROI's of the neurons can be plotted on top of the cellmaps. This can be done by selecting the menu *Annotations* and your preferred labelling analysis pipeline. To differ between the different methods the ROI's are plotted in different color maps (figure 4.5). The color maps can easily be changed over the setting section in the code. Once the spatial annotations are loaded they can be Hidden by selecting the same *Show / Hide -method name-* again. This helps to keep an overview.

Depending on the settings section in the code, the plotted ROI's are movable and selectable. The following main actions can be taken:

1. Delete a detected neuron
2. Edit the shape of a neuron
3. Add a new neuron by drawing its shape

Deleting one or multiple neurons is possible by selecting the neuron(s) and pressing delete. The neuron and its corresponding temporal traces are permanently deleted. Moreover, it is possible to move the single coordinates of a detected neuron to adjust its shape. Finally, if a neuron was not detected by

any previous method a new neuron can be drawn over selecting the menu *Edit Neurons* and *Draw new*. As many single points as needed can be added to shape the contour of the neuron. Figure 4.3 displays a self drawn neuron shape.

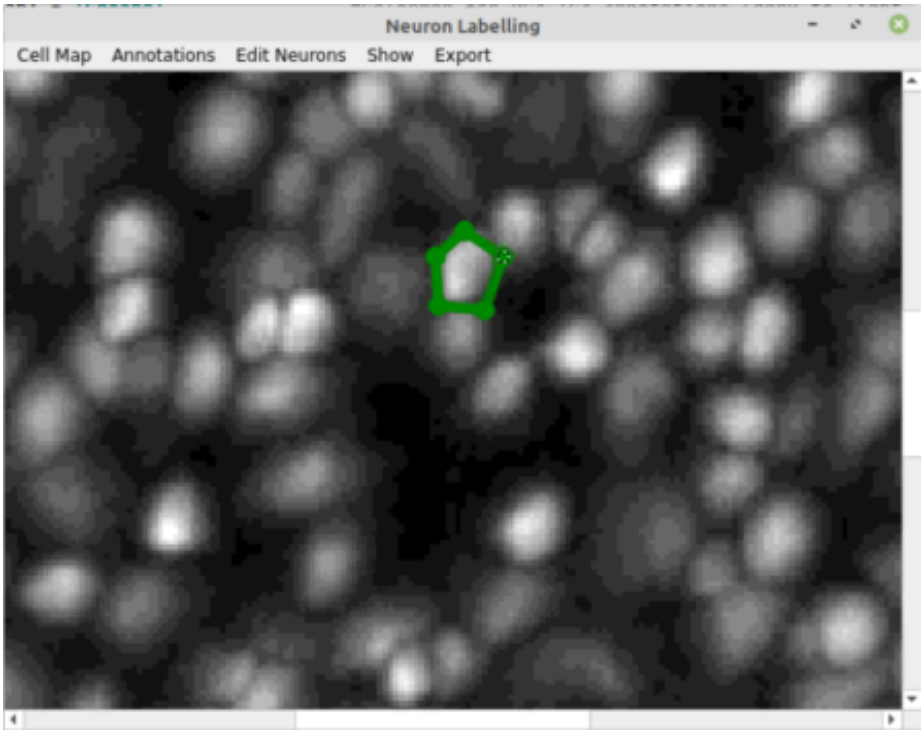

Figure 4.3: A self drawn neuron in the annotation window

One frequently recurring problem is shown in figure 4.4. Often detected neurons are overlapping, sometimes fully covering a neuron plotted below. In this case it is not possible to select the neuron below. To work around this problem, one can simply select the top neuron and use a keyboard shortcut to move it down a layer. This way, neurons below can be accessed and edited. This action can likewise be performed the other way around: a neuron lying below can also be moved up, even if this action is not frequently required.

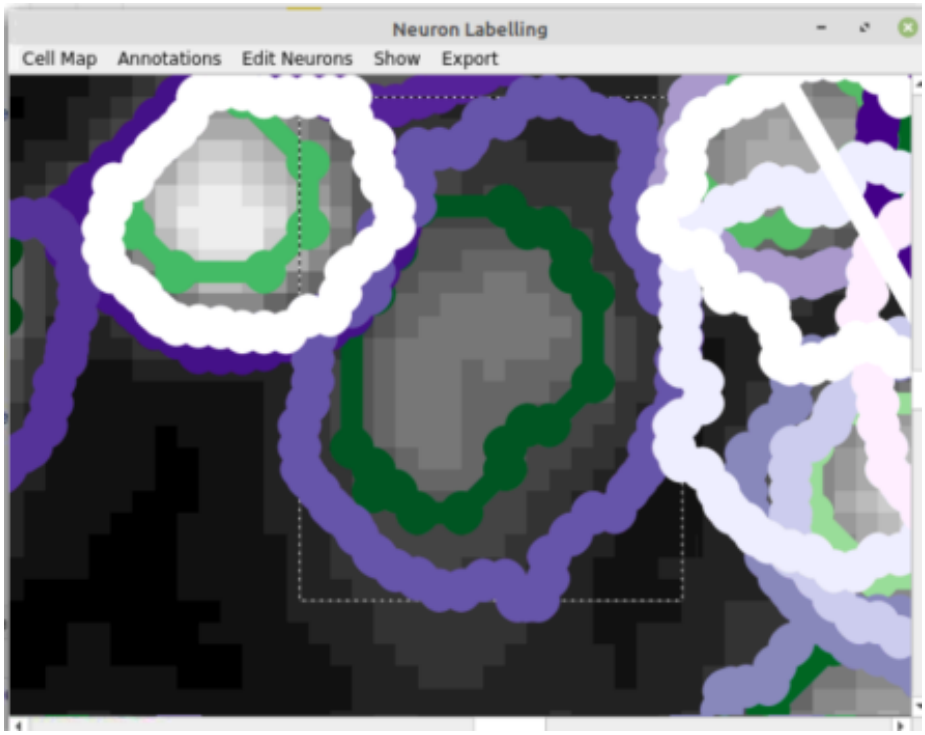

Figure 4.4: A neuron ROI fully covering another neuron ROI plotted below

4.1.3 Background

The annotation background can be set to the produced cellmap of any added method. The annotations of both the PCA-ICA and the CNMF-E algorithms with both possible backgrounds are shown in figure 4.5. The action can be triggered by a keyboard shortcut or by the manual selection of the menu.

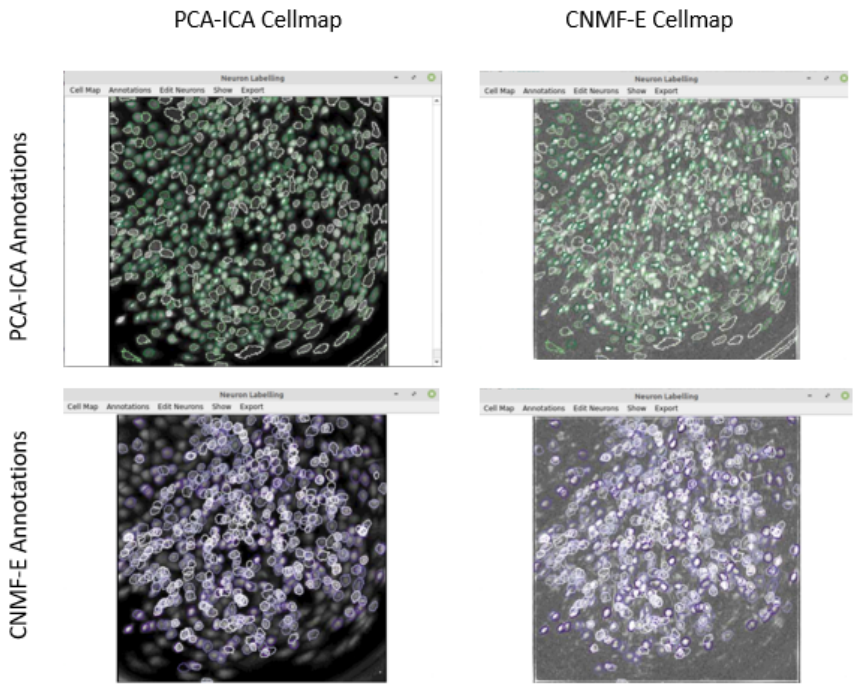

Figure 4.5: PCA-ICA (top) and CNMF-E Annotations (bottom) shown above the PCA-ICA cellmap (left) and CNMF-E cellmap (right)

4.1.4 Zoom

Since neurons and neuronal activity are hard to detect a zoom-in and zoom-out function which can be triggered by keyboard shortcuts help visualizing details. With scroll bars on the side of the window it is possible to navigate through the 500x500 pixels. Other features update their zoom accordingly to the main window. When changing the background the zoom and center coordinates are preserved. Additionally the movie window 4.1.5 can be updated as well.

4.1.5 Movie

To sufficiently understand whether a detected neuron is a false or true positive the pre-processed movie can be used as an aid, since neurons light up in the movie. Under the menu category *Show Movie* can be selected to open a new window which is showing the movie (figure 4.6). Once the annotations of a method are loaded in the main annotation window, they can also be loaded in the movie window via the menu category *Show* and the selection of the preferred method. Once the neuron annotations in the main window have been edited (deleting shapes, changing shapes), the annotations in the movie window can

be updated by simply selecting *Hide* and *Show* once more. Furthermore, as described in chapter 4.1.4 the zoom and center applied in the main window can be transferred to the movie window, thereby aiding navigation and orientation. Figure 4.7 shows both windows with loaded annotations and applied zoom.

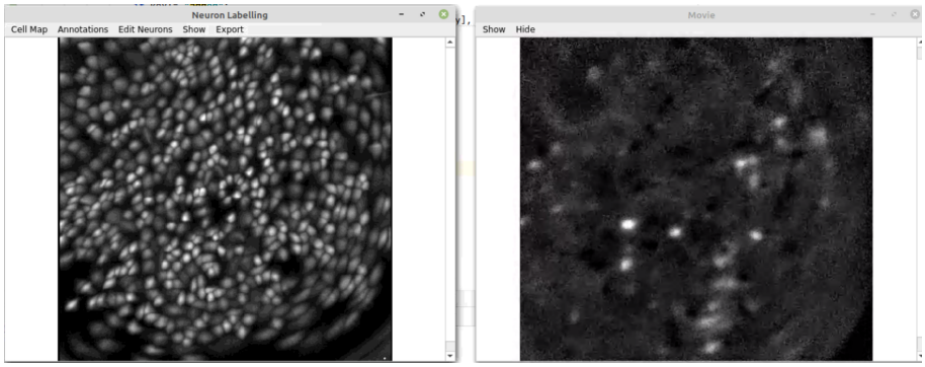

Figure 4.6: Main Annotation Window and Movie Window

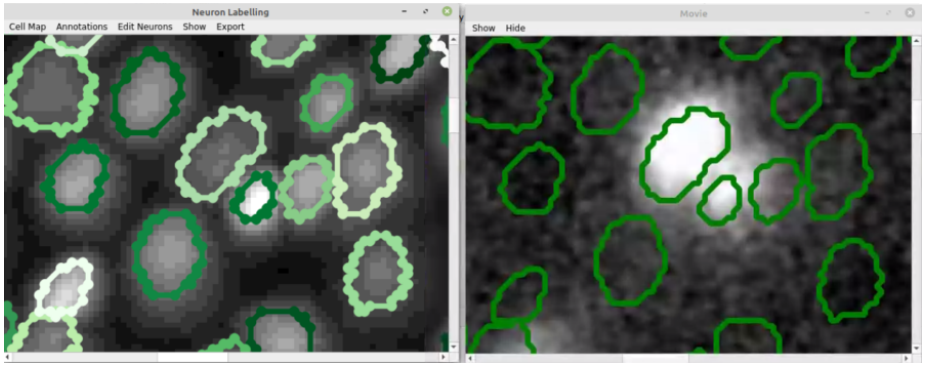

Figure 4.7: Main Annotation Window and Movie Window with loaded Neuron Annotations and applied Zoom

4.1.6 Calcium Traces

The associated calcium traces are another aid for the accurate labelling of neurons. The unsupervised algorithms also use the ratio of peak to signal idle state and trace fluctuations as an indicator for whether a detected seed is a neuron [12]. To visualize the calcium traces of selected neurons the menu *Show* and sub-menu *Traces* can be selected. Next, the trace window can be updated by a keyboard shortcut. This way, neuron ROI's of interest can be selected and their traces can be compared (figure 4.8). From figure 4.8 can be seen that the conversion factor of the arbitrary units is $\frac{Calcium\ Intensity\ PCA\text{-}ICA}{Calcium\ Intensity\ CNMF\text{-}E} = \frac{1}{100}$.

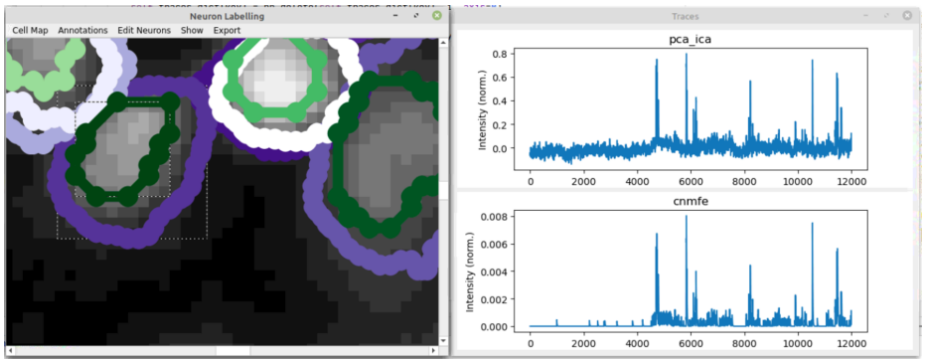

Figure 4.8: Main Annotation Window and Traces Window with two selected neuron ROI's and their displayed calcium traces

4.1.7 Export Results

Finally, once the labelling is completed, the merged and updated annotations can be exported to a new `.mat` file. The `.mat` file will contain the following information about all remaining neurons:

1. Coordinates of all neurons ROI's
2. Binary Images of all neurons ROI's
3. Calcium Traces of all neurons (besides self-drawn)
4. Cellmaps of the different Input Labelling algorithms

4.2 Implementation

The Graphical User Interface is build in Python 3.7 with the python library `PyQt5`. The project was developed object-oriented. The code is split into two `.py` files: `load_data.py` and `main.py`, where the main file contains almost all functionalities and the load file simply holds loading functions for annotation data.

4.2.1 Windows

Windows are either derived from `QtWidget` or `QtWidgets.QMainWindow`. A main window can have a `QGraphics.View` and a `QGraphics.Scene`, where the scene is managing all items and the view is acting similarly to a camera filming the scene. If a scene and a view are implemented, more complex actions such as overriding functions and managing multiple items with various functionalities are possible. In this project the main window and the video window have a scene and view object as a class instance. The trace window on the other hand is a simple widget as it does not provide complex functionalities. If a window has a scene manager object most functions are outsourced to the `QGraphics.Scene` derived scene class. In case of the main window most functionalities are implemented in the `AnnotationScene` class. The `AnnotationScene` holds all information about extracted neuronal signals, transforms coordinates into objects of the `PolygonAnnotation` class, adds and deletes them from the scene. The `AnnotationView` class manages view functionalities such as zoom and center related functions. The main window and video window both have objects of the same `AnnotationView` class since both depend on the same zoom functions.

The main window is created automatically as soon as `main.py` is executed. In this class the menus and most shortcuts are created. Thus, almost all actions are triggered from here. If the user selects to show a new window such as the trace window or the video window the corresponding objects are created as instances of the main window class and forwards its own scene manager object, so that the new windows have access to all the information stored in the annotation scene such as the extracted neuronal signals. Thus, the trace window is able to access the calcium traces and has information about which items in the scene are selected. Likewise the video window is able to plot ROI's of neurons and can update these in case modifications were made in the main window.

4.2.2 Annotations

Although all annotation information is stored in the `AnnotationScene`, the displayed polygon objects on the surface of the main window are objects from another essential class named `PolygonAnnotation` which is derived from a `QGraphicsPolygonItem`. Each polygon object in turn has several `GripItems`, which represent the coordinates of the neuron's contours. In both the `GripItem` class and the `PolygonAnnotation` class basic functions are overridden. This allows more functionalities to be implemented, for example, triggered by a

`hoverEvent` or an `itemChange`. This way neuron shapes and their single coordinates can be moved and changed.

4.2.3 Actions

An action can be triggered by a mouse click on a button or space in a window or a keyboard shortcut. The menus and most keyboard shortcuts are implemented in the main window class. An exception is the shortcut for applying zoom. As this is closely connected to the functionality of the view it is implemented in the `QGraphics.View` class. Some actions are dependent from other more global actions. For example, a mouse click on the main window surface is interpreted differently if the drawing a neuron action is enabled. In this case a `QPoint` will be added at to the scene, whereas if no drawing action is enabled a mouse click could select an item in the scene or simply trigger nothing. Therefore, some of the actions set toggling enumerations which can provide information about whether the action is enabled or not.

4.3 User Manual

4.3.1 Dependencies

The following packages have to be installed:

- PyQt5
- matplotlib
- scipy
- PIL
- numpy
- cv2
- h5py

Furthermore, GStreamer has to be installed in order to play the video.

4.3.2 Loading data

Once all dependencies are installed, paths to select the different annotation and calcium imaging data files need to be set in the beginning of the file. For each method, the `.mat` file containing information about the neuronal signals and the extracted cellmaps must be accessible. For the PCA-ICA method binary masks are sufficient. Coordinates of the contours of binary masks are extracted in the `load_data.py` file. The project was build in such a way that a new method can easily be added. The following actions need to be taken:

1. Add paths and edit the `load_annotations` function in the `Annotation_Scene` class

2. If temporal and spatial information is organized differently as in the PCA-ICA or CNMF-E `.mat` files, add functions in the `load_data.py` file

3. Add key to the dictionaries containing information about the annotations

4.3.3 Settings

A settings section in the beginning of the code allows the user to select different settings before the labelling process is started. Color maps for the different neuronal extraction algorithms can be set. Additionally, the user can decide whether an annotation and / or its single coordinates should be movable. Further settings can be added.

4.3.4 Keyboard Shortcuts

In table 2 all possible keyboard shortcut actions are visible.

Table 2: Overview of Keyboard Shortcuts

Keyboard Shortcut	Function	Menu Option
Ctrl and +	Zoom in	-
Ctrl and -	Zoom out	-
C	Change Background to CNMF-E cellmap	Cell Map → CNMF-E
P	Change Background to PCA-ICA cellmap	Cell Map → PCA-ICA
Del	Delete selected neurons	-
B	Move selected neurons one layer down	-
T	Move selected neurons one layer up	-
V	Update video zoom and center	-
U	Update trace window	-
D	Start drawing a neuron	Edit Neurons → Draw new
Esc	When drawing a neuron, finish neuron drawing action	-