Rhythm 1.2 GUI user guide

by Roman Pryamonosov, Alexander Zolotarev, Roman Syunyaev

# Getting Started

Be sure that you have Matlab version 2017 or greater.

Download Rhythm 1.2 to your project directory from GitHub via link: https://github.com/optocardiography/Rhythm-1.2

Open rhythm.m with Matlab and go to Home -> Set Path. Add your project directory via “Add with Subfolders …”, then “Save” and “Close”.

Run rhythm.m.

# GUI

Main window of Rhythm consists of 3 parts (see figure 1):

1. Toolbars on the left
2. Four movie screens on the center
3. Signal windows on the right

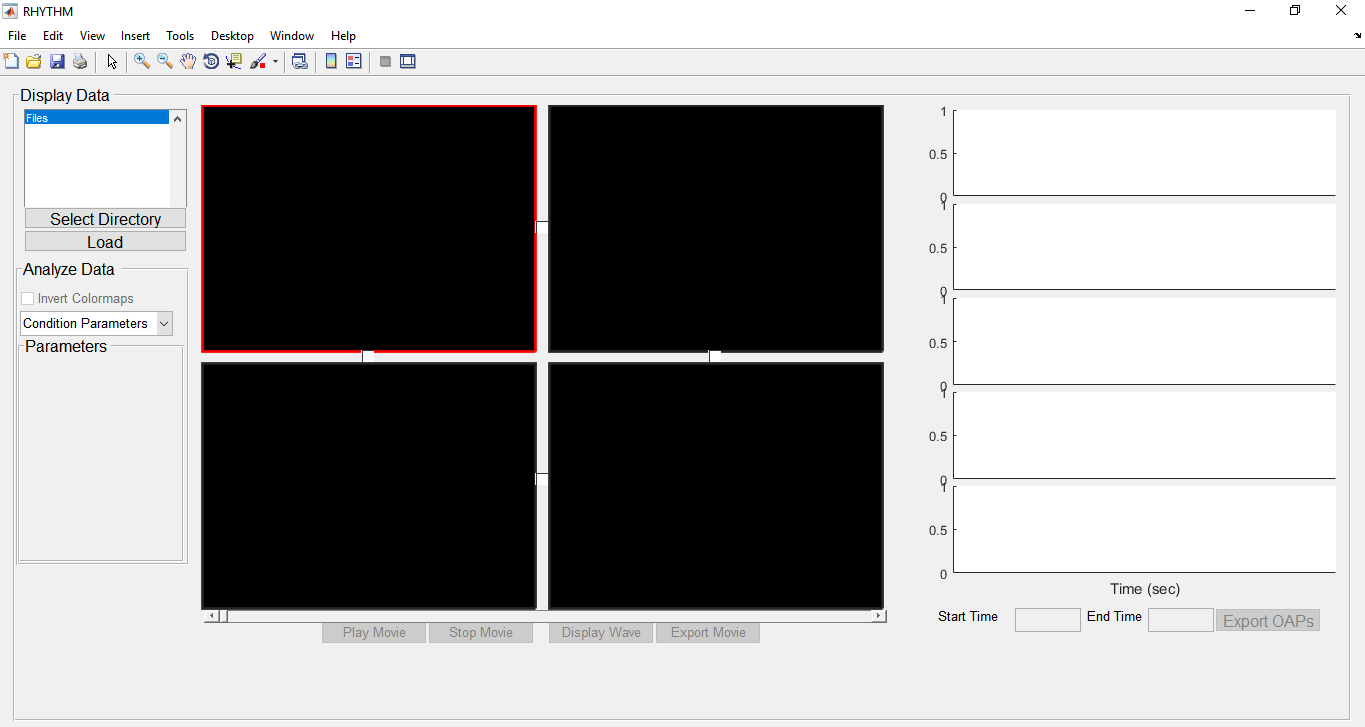


Figure 1. Start interface of Rhythm. Toolbars on the right, four movie screens on the center, signal screens on the right.

# Toolbars

Use button “Select Directory” on the top of toolbar menu to select directory with experimental data. You will see the list of available files in the current directory. Select a file and click “Load” to load experimental data and visualize it on the selected movie screen (marked by red boundaries). Load different data file to another movie screen if necessary.

Menu “Analyze Data” on the bottom has popup menu containing several functions for signal processing (Condition Parameters, Conduction Velocity, Activation Map, etc.). Once data is loaded to movie screen(s) you can select signal processing function, that you are interested in. After that you will see parameters for selected processing method. See several examples on figure 2. “Save” button (diskette icon) allows user to export signal processing results as a matlab figure.

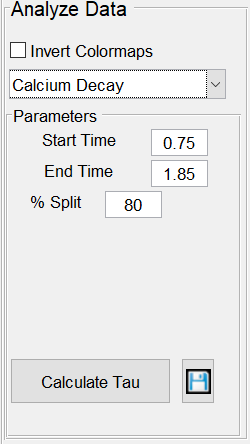
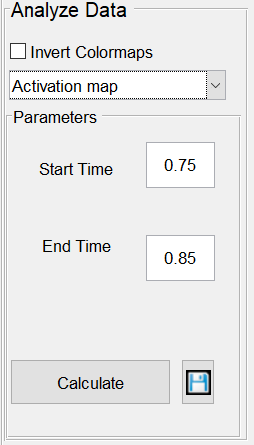


Figure 2. Examples of “Analyze Data” menu.

# Movie screens

Four movie screens allow user to upload several data files independently (NB: all loaded data must have a common frame rate). The selected screen has red boundaries. Single-click selects screen, double-click zooms in the selected screen and provides the expanded selected screen.

Once data is loaded, buttons Play Movie, Stop Movie, Display Wave and Export Movie become active. “Play movie” and “Stop Movie” buttons are used to visualize signals recorded by camera. Slider below movie screens and the red sweep bar on signal screens show current time of the movie.

Use “Display Wave” button to set marker on the desirable point of a movie screen. The signal from selected points are visualized on the signal screen (right-hand-side of the interface, see below). You can set up to 5 different markers. After setting 5 markers, new Display Wave calls overwrite old markers. You can use different sets of markers for each screen, while only signals from the currently active screen are depicted on the signal screen. Left-click on the movie screen to activate it. For example, figure 3 shows two datasets with 2 and 4 markers correspondingly. Markers can be grabbed by single-click and dragged to a different position of its movie screen.

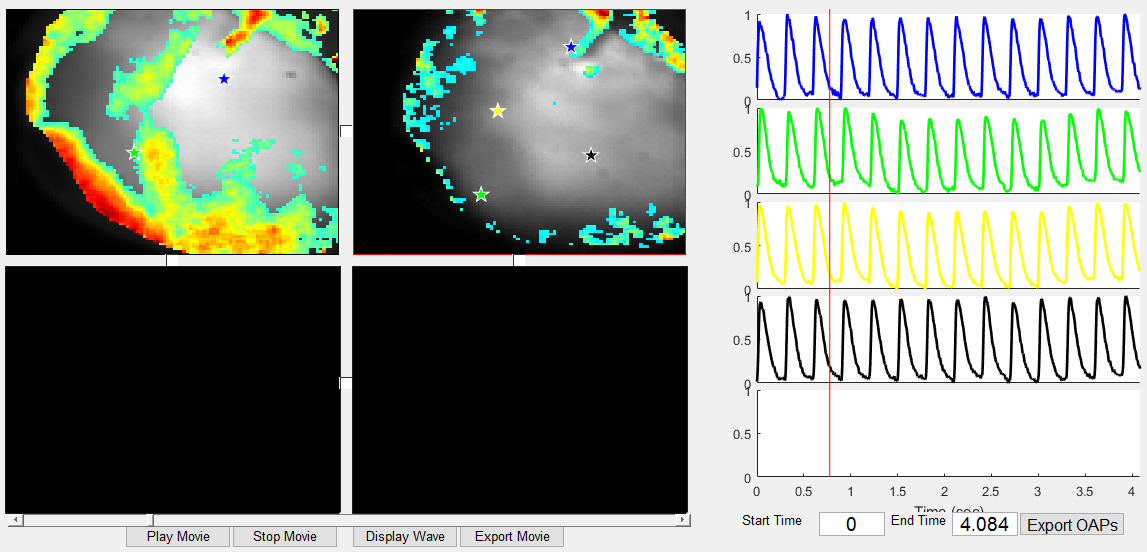


Figure 3. Several markers are set on different screens. Signal waves in marker points are shown on the signal screens for selected screen.

Screen synchronization

Several data files obtained from the same point of view may be synchronized. User can synchronize uploaded data by setting checkboxes between the movie screens. Once several movie screens are synchronized, they share markers. Marker, placed on one of synchronized screens, is applied to all the screens from that group. User can drag group markers to a different position as usual. Figure 4 shows two synchronized movie screens 1 and 2, and one separate screen 3. Screens 1 and 2 have two group markers, while screen 3 has its own set of markers. Signal waves are drawn for a selected group of synchronized screens.

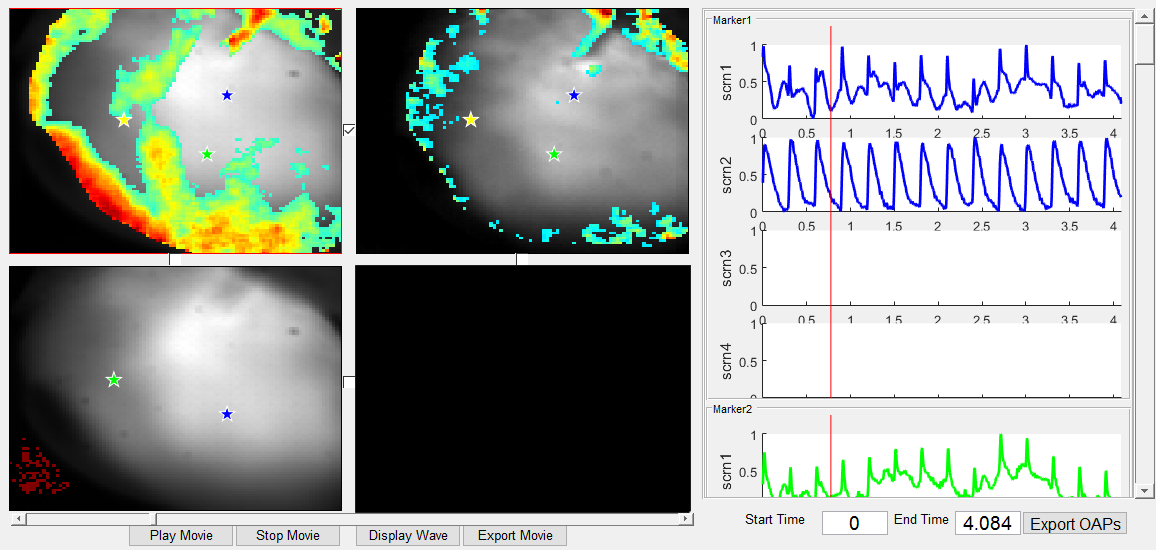


Figure 4. Example of the two synchronized movie screens and one screen with no group. Signals on the right are shown for the screen group, that contains the selected screen.

# Signal screens

Signal screens on the right show time-dependent signal waves, probed in the position of markers. Signal waves have the same colors as their markers.

Signal screens display signal waves in two modes: waves of a particular signal screen and waves of a group of synchronized screens. In the first case (see figure 3) there are 5 signal screens (one for each marker). In the second case (see figure 4) there is a group of four signals (one for each movie screen) for each marker. Vertical slider allows user to scroll down signal screens.

Red vertical line represents the current time. User can zoom in/out in time, specifying “Start Time” and “End time”. Note that start time should be non-negative value.

Button “Export OAPS” exports signal waves to a new window that can be used to save data as image. In synchronized movie screen mode, the new window organized as follows: columns stand for screens, rows for markers (see figure 5).

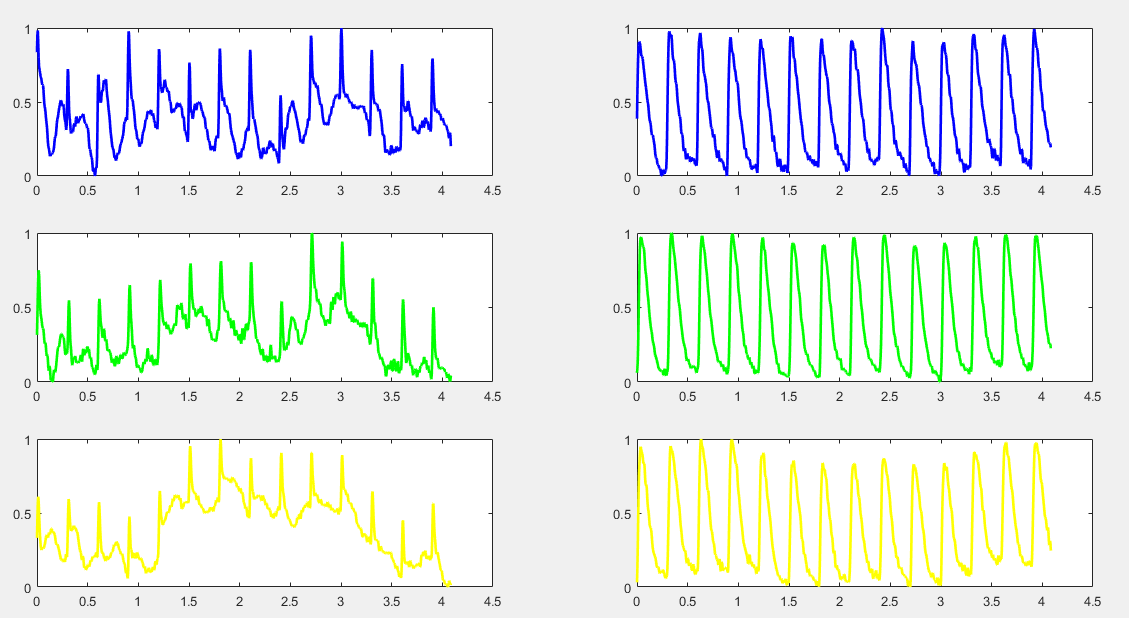


Figure 5. Export OAPs window. Example for 2 linked movie screens with 3 markers on them.

# Using signal conditioning and analysis functions

There is the default version of Rhythm-1.2, and it has only Signal Conditioning block and 5 separate functions: Activation map, Conduction Velocity map, Action Potential Duration map, Rise Time map and Calcium Decay map. If user wants to add any other map or calculation to this code, this program provides an easy way to do it. We describe all steps for it in “Add new user-implemented analysis functions” section.

User should prepare the raw data before using any function. To understand any possibility for preparing, please read “Signal conditioning” section. This program offers following options for conditioning: Removing Background, Binning, Filtering, Removing Drift, Normalization and Signal Inverting.

## Signal conditioning

The Condition Signals panel consists of five check boxes (Remove Background, Bin, Filter, Remove Drift, Normalize, Invert signal) and one Apply button. The check boxes may be selected in any combination, but the conditions will be applied in the order they appear after *Apply* is clicked. Only the original data will be processed.

#### Remove Background

This function segments the black and white image of the tissue taken by the CMOS camera and detects pixels below the BG Threshold value. It then uses the EX Threshold to remove groups of pixels less than EX of the total image size. EX Threshold and BG Threshold values are set to values between 0 and 1. The figures below demonstrate the effect of removing the background of the original signal.

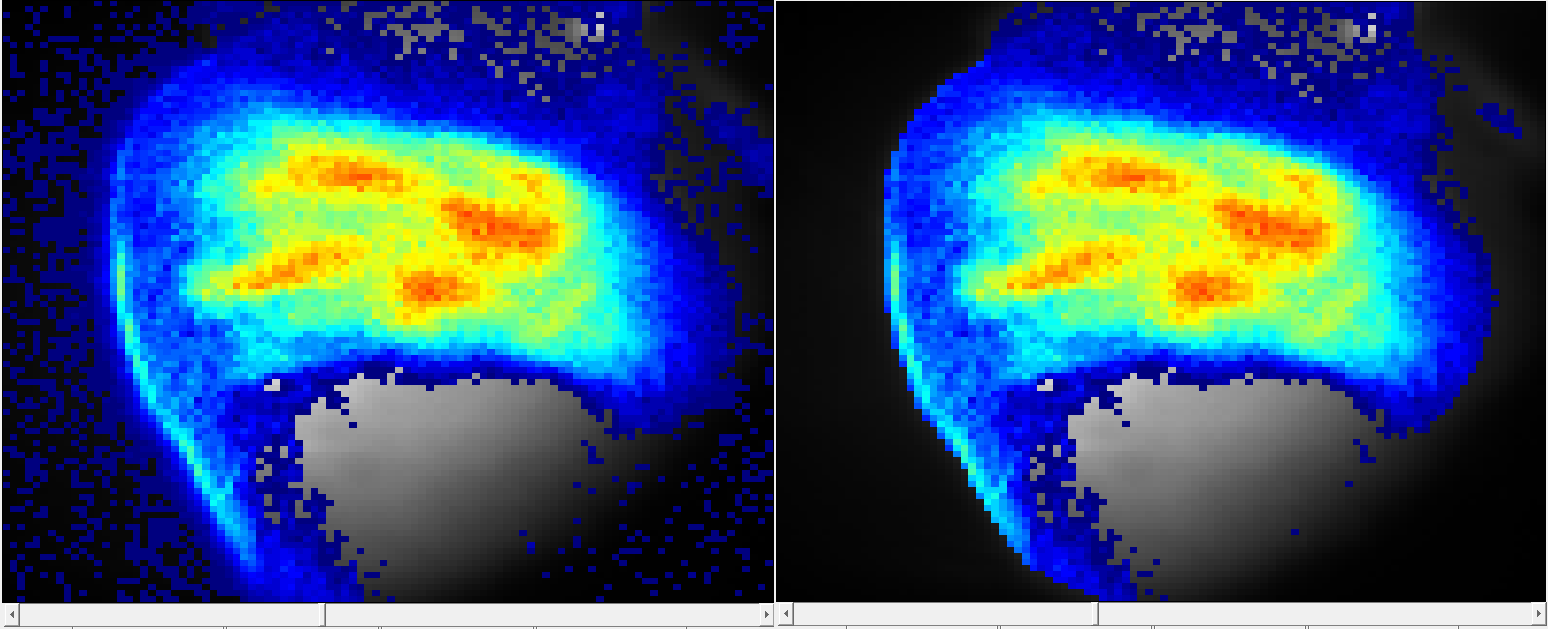


Figure 6. Remove Background

#### Bin

The Bin function sums the amplitudes of adjacent pixels, divides the sum by the number of pixels, and puts the average value in the center pixel. Increasing the bin size increases the smoothness. Decreasing the bin size increases sharpness.

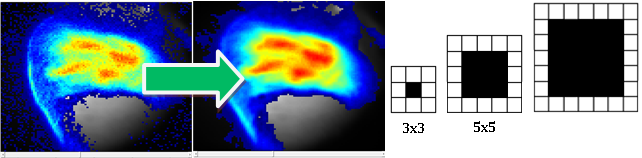


Figure 7. Binning

#### Filter

The Filter function operates zero-­‐phase forward and reverse digital filtering with an nth order band pass filter between a low and high passband threshold frequency. By clicking the drop-­‐down menu next to Filter, the high band of the filter can be adjusted to 50, 75, 100 and 150 Hz. All filters are implemented with the Parks-McClellan-Remez Exchange algorithm and made to be zero-­‐phase.

#### Remove drift

The Remove Drift function adjusts a slanted waveform to level off the baseline.

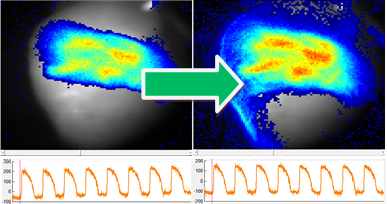


Figure 8. Drift removal

#### Normalize

The Normalize function adjusts the data set to normalize data points to be within 0 and 1. Normalization should be performed once the background noise pixels are removed. Below is an example.

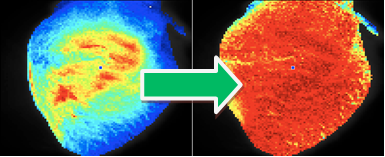




Figure 9. Normalize signal.

#### Inverse data

The Inverse Function has two possible variables. If user wants to inverse signal upside down, it’s necessary to check “Y” (it is turn around Y axe). To inverse signal left to right, check “X” (it is turn around X axe). To inverse signal around X and Y axes, check both “X” and “Y”.

## Activation map

The Activation Map function plots the activation map for the time duration as specified in the Start Time and End Time entries. This interval is restricted by green sweep bars in the signal Screens right after entering numbers in Start Time and End Time entries. For view activation map in selected screen, use “Calculate” button. In order to view this map as a separate Matlab figure, click on the “Save” icon. User can save this map in any format provided by Matlab (for example .fig, .jpg, .png, etc.).

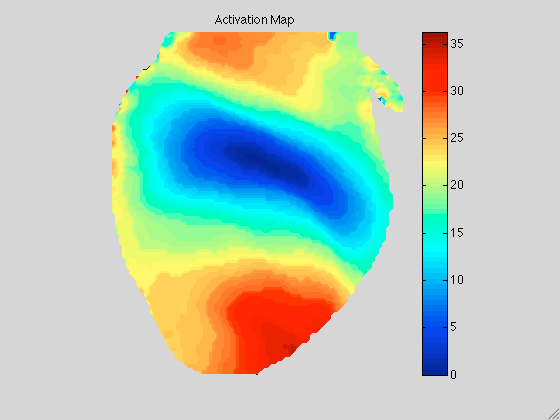


Figure 10. Activation map.

## Action potential duration map.

The Start Time and End Time entries restrict the data into a specific time interval, reflected by sweep bars in the Signal Screen. The Min APD and Max APD entries allow the user to control the range of action potential duration in milliseconds, based on the data being analyzed. The entries must be numeric values greater than zero. The %APD button allows control over analysis of the time duration, from the maximum upstroke velocity to the first point that is a specified percentage of the maximum value after the peak as depicted in figure 11. Upon click on the “Mapping” button APD map appears on the selected screen. Pop-up figure is a histogram of APDs displayed in the map. In order to export this map as a Matlab figure, click “Save” icon.

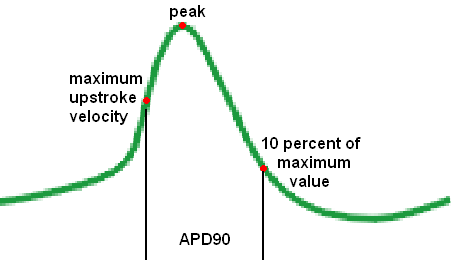
****

Figure 11. The %APD entry allows control over analysis of the time duration. For example, 0.9 value in the entry results in APD90 calculation as depicted in the figure.

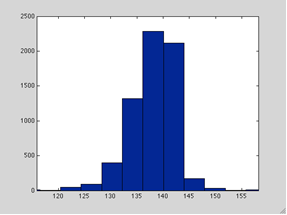
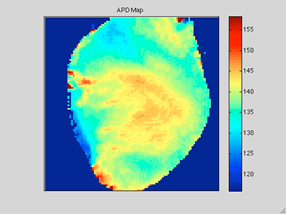


Figure 12. APD map.

Upon clicking “Regional APD” select rectangular region of interest on the selected screen with the cursor. Three variables are displayed in the Matlab Command Window and in the Statistics panel. These are APD mean (Mean in Statistics), APD median (Median in Statistics) and the standard deviation of the action potential (S.D. in Statistics). #Members and Angle values in Statistics panel continue to be empty.

## Conduction velocity map

The function calculates conduction velocity (CV). CV is calculated as local gradient of activation time. Edit start and end times entries to define the time interval. This interval is reflected by blue sweep bars in the signal Screens right after entering numbers in Start Time and End Time entries.

Button “Generate Vec. Map” provides two options: calculation statistical parameters within selected region of interest and imaging Conduction Velocity map. There are two possibilities for imaging: first - only vectors of CV and second – vectors of CV plus activation map as a background. To select second option, check “Contour”.

After clicking “Generate Vec. Map”, select a rectangular region of interest on the selected screen with the cursor. Vector map of CV with or without activation map (refers to checking “Contour”) is depicted on the screen afterwards. Five numbers are printed in Statistics panel. These are median conduction velocity, mean conduction velocity, standard deviation of the CV, the total number of vectors included in calculation and mean vector angle.

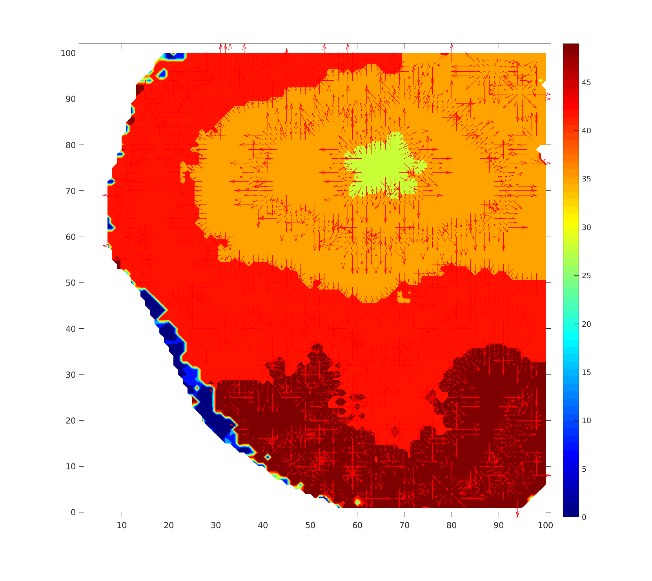


Figure 13. CV map.

Sometimes it’s necessary to calculate CV along one direction (e.g. longitudinal or transverse CV). This option opens only after completing vector map. To do this option, click “Draw Line”. Then select start and end points by left-click on the starting point of the direction of interest, right-click on the ending point afterwards. Then click “Calculate CV”. It calculates average CV in a rectangle within 5 pixels of the line and with CV direction within 15° of the selected line direction. Selected vectors are viewed by blue color, also user can see prohibited vectors by red color.

Click “Save” icon in order to export CV map as a Matlab figure. User can save this map in any format provided by Matlab (for example .fig, .jpg, .png, etc.)

## Rise time

The function calculates depolarization time. The Start Time and End Time entries restrict the data into a specific time interval. The Rise Time is calculated as a time from the time of %Start fraction of depolarization to the time of %End fraction of depolarization.

To provide map and calculations, click “Calculate” button and select the rectangular region of interest on the screen. Rise Time distribution within region of interest and histogram are displayed. Also four numbers are printed in Statistics panel. These are median depolarization time, mean depolarization time, standard deviation of the depolarization time, the total number of points. Angle value in Statistics panel continues to be empty.

“Save” button exports the Rise Time map as a Matlab figure. User can save this map in any format provided by Matlab (for example .fig, .jpg, .png, etc.).

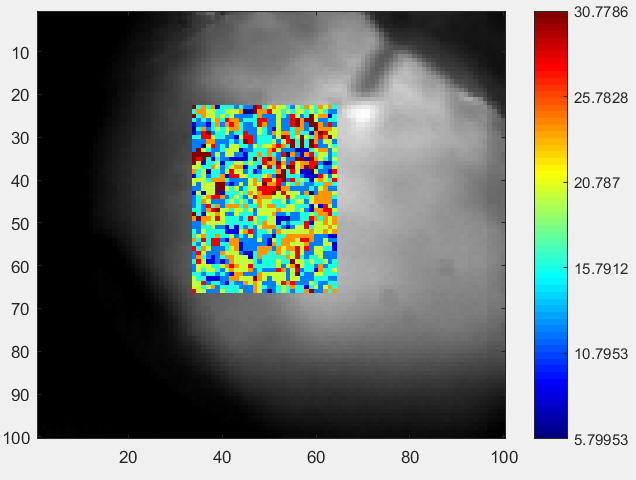


Figure 13. Rise Time map.

## Calcium decay

This function calculates the relaxation time of calcium transients. The Start Time and End Time entries restrict the data into a specific time interval. Calcium transients from the “%split” fraction of the calcium amplitude to the 10% of calcium transients amplitude are fitted with exponential functions.

Click on “Calculate Tau” button and select the rectangular region of interest. Relaxation time distribution is depicted on the screen afterwards. Also four numbers are printed in Statistics panel. These are median time, mean time, standard deviation of the time, the total number of points. Angle value in Statistics panel continues to be empty.

“Save” button exports the Calcium decay map as a Matlab figure.

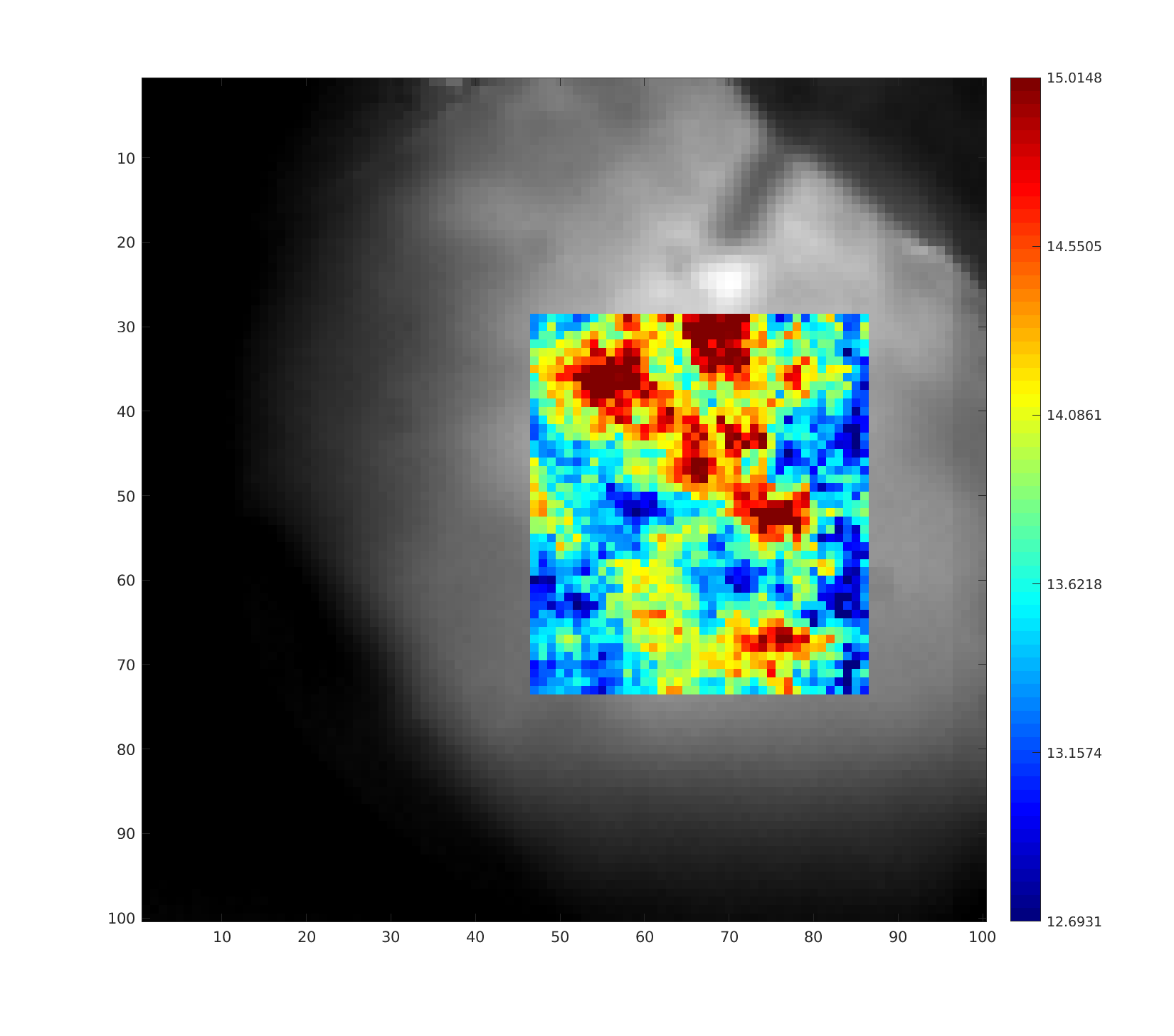


Figure 14. Calcium decay map.

# Add new user-implemented analysis functions

One can add new functional with the minimal interaction with rhythm.m with the following next steps.

1. Create a name for your functional

In rhythm.m find the line where map\_popup variable is initialized:

map\_popup = uicontrol('Parent',anal\_data,'Style','popupmenu','FontSize',10,...

'String',{'Condition Parameters','CV map',...

'Activation map', 'APD\CaT map', 'Rise Time', 'Calcium Decay'},...

'Position',[3 320 140 25], 'Callback',{@mapPopUp\_callback});

That line defines the set of toolbars in the “Analyze Data” menu. Add a new name into a ‘String’ field into the uicontrol, for instance, 'Example Function':

map\_popup = uicontrol('Parent',anal\_data,'Style','popupmenu','FontSize',10,...

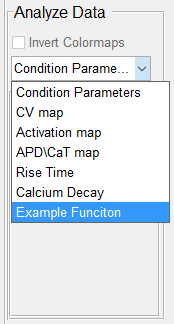
'String',{'Condition Parameters', 'CV map',...

'Activation map', 'APD\CaT map', 'Rise Time', 'Calcium Decay',...

’Example Function’},...

'Position',[3 320 140 25], 'Callback',{@mapPopUp\_callback});

It will allow you to see the name you have added (“Example Function” in our example) in popup menu:



1. Bind your add-on functionality to GUI

In rhythm.m find a definition of the function mapPopUp\_callback(~,~) and add a corresponding case:

function mapPopUp\_callback(~,~)

switch get(map\_popup,'Value')

...

case 7

GUI\_ExampleFunction (map, handles, f);

end

...

Note that all GUI functions share at least 3 essential arguments. The first argument map stands for a panel to draw on. The second argument handles is the handles class of main rhythm.m program (for more details see the handles class implemented in rhythmHandles.m). The third argument f is the main figure of the program window. One can add auxiliary input arguments.

## Tip:

Do not use input arguments for values you need to save. Use handles fields to let the whole program know that something is saved or changed. To this end, go to rhythmHandles.m file and add a new field.

1. Create GUI panel for your add-on

Write your own GUI functional using the template GUI\_example.m. Create new m-file GUI\_ExampleFunction.m for your GUI and copy everything from GUI\_example.m. As mentioned above the function for your GUI must have at least three fixed arguments:

GUI\_ExampleFunction (map, handles, f)

Add any matlab GUI structures, specifying the ‘Parent’ field as the map. For example, add time edit functional with the following line:

timeEdit = uicontrol('Parent', map,...

'Style','edit','FontSize',10, ...

'Units','normalized',...

'Position',[0.6 0.8 .3 .15], ...

'Callback', @timeEdit\_callback);

Use figure f as a usual figure. For example in GUI\_ActivationMap.m we in drawTimeLines function we set a field ‘CurrentAxes’ of the main figure to some sweepBar of the main GUI window.

function drawTimeLines(val\_start, val\_end, handles, f)

...

set(f,'CurrentAxes',handles.sweepBar)

...

Inside of your GUI structures you can use handles to access camera data on the active screen, screens and so on:

* Use handles.activeCamData to get the data depicted on the active screen.
* Use handles.activeCamData.screen to get the screen itself.
* Use handles.activeCamData.cmosData to get the camera data you preprocessed before and handles.activeCamData.rawData to get the raw camera data.

See the handles class camData.m for more details.

The next step is essential to provide stable visualization. After you draw your map on the active screen, set handles.activeCamData.drawMap to 1 for correct map drawing. For example, in file GUI\_ActivationMap.m we use aMap() fuction to draw something on the active screen and then set handles.activeCamData.drawMap to 1:

aMap(handles.activeCamData.cmosData,handles.a\_start,handles.a\_end,...

handles.activeCamData.Fs,handles.activeCamData.cmap,...

handles.activeCamData.screen, handles);

handles.activeCamData.drawMap = 1;

Tips:

* It is strongly recommended to use normalized coordinates.
* You can find other GUI examples already implemented in rhythm. See m-files started with “GUI\_”: GUI\_ActivationMap.m, GUI\_CalciumDecay.m, etc.