Astral – Manual

Contents

[Introduction 1](#_Toc60254871)

[Technical Specification 1](#_Toc60254872)

[Installation and Running 5](#_Toc60254873)

[Usage 6](#_Toc60254874)

[Filename 9](#_Toc60254875)

[Intersect\_threshold 9](#_Toc60254876)

[Standard\_deviation\_height\_range 11](#_Toc60254877)

[Standard\_deviation\_threshold 11](#_Toc60254878)

[Standard\_deviation\_width\_range 11](#_Toc60254879)

[Tolerance\_t 11](#_Toc60254880)

[Tolerance\_xy 12](#_Toc60254881)

[Use\_watershed 14](#_Toc60254882)

[Volume\_threshold 15](#_Toc60254883)

# Introduction

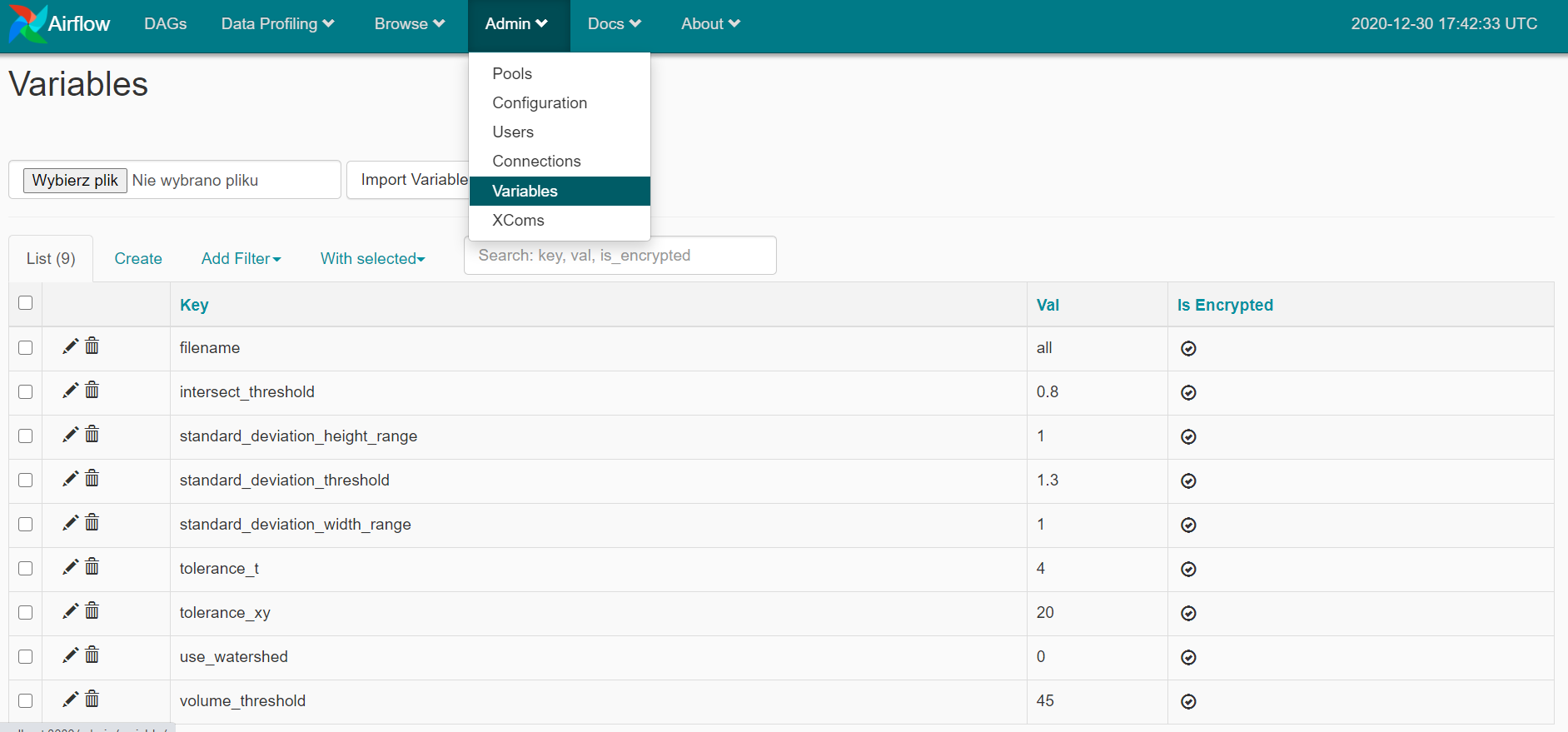
Astral is a tool for analyzing and interpreting calcium signalling from a microscopic timelapse imaging. The goal of Astral is to create an easily extensible system for such data. As such experiments produce complex data, Astral was created for detection, segmentation, and analysis of calcium events inside timelapse data.

There are couple of parameters available in Astral. You can see a full list of the parameters with their description below:

# Technical Specification

|  |  |  |
| --- | --- | --- |
| Parameter Name | Description | Default Value |
| [Filename](#_Filename) | A filename of a timelapse sequence to be processed from the data directory. Leave ‘all’ if all the files should be processed | All |
| [Intersect\_threshold](#_Intersect_threshold) | How much should the calcium waves overlap in their z-projection to be treated as repeats. | 0.8 |
| [Standard\_deviation\_height\_range](#_Standard_deviation_height_range) | How many adjacent pixels in height dimension to take into account during std calculation | 1 |
| [Standard\_deviation\_threshold](#_Standard_deviation_threshold) | Standard deviation threshold for detecting calcium events | 1.3 |
| [Standard\_deviation\_width\_range](#_Standard_deviation_width_range) | How many adjacent pixels in width dimension to take into account during std calculation | 1 |
| [Tolerance\_t](#_Tolerance_t) | A maximum distance between subsequent calcium waves to treat them as neighbours | 4 |
| [Tolerance\_xy](#_Tolerance_xy) | A maximum distance in xy plane to treat two different calcium waves as neighbours | 20 |
| [Use\_watershed](#_Use_watershed) | Whether during mask generation, a Watershed algorithm for splitting adjacent calcium waves should be used | 1 |
| [Volume\_threshold](#_Volume_threshold) | Volume threshold in total number of pixels to treat a shape as a calcium wave. Calcium waves below volume\_threshold will be discarded from the analysis. | 45 |

Those parameters can be set in the Variables section in the Admin tab:



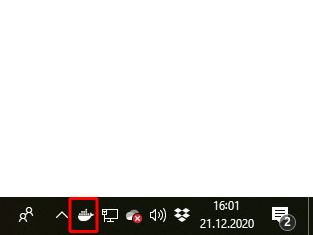
To change the parameter value, you may use a Pen icon next to the parameter name.

It is possible to override the default parameter values. These values are loaded from the **variables.json** file, located in the parent directory of the application. Modifying the values inside the **variables.json** file will change the default values of the application from the next startup.

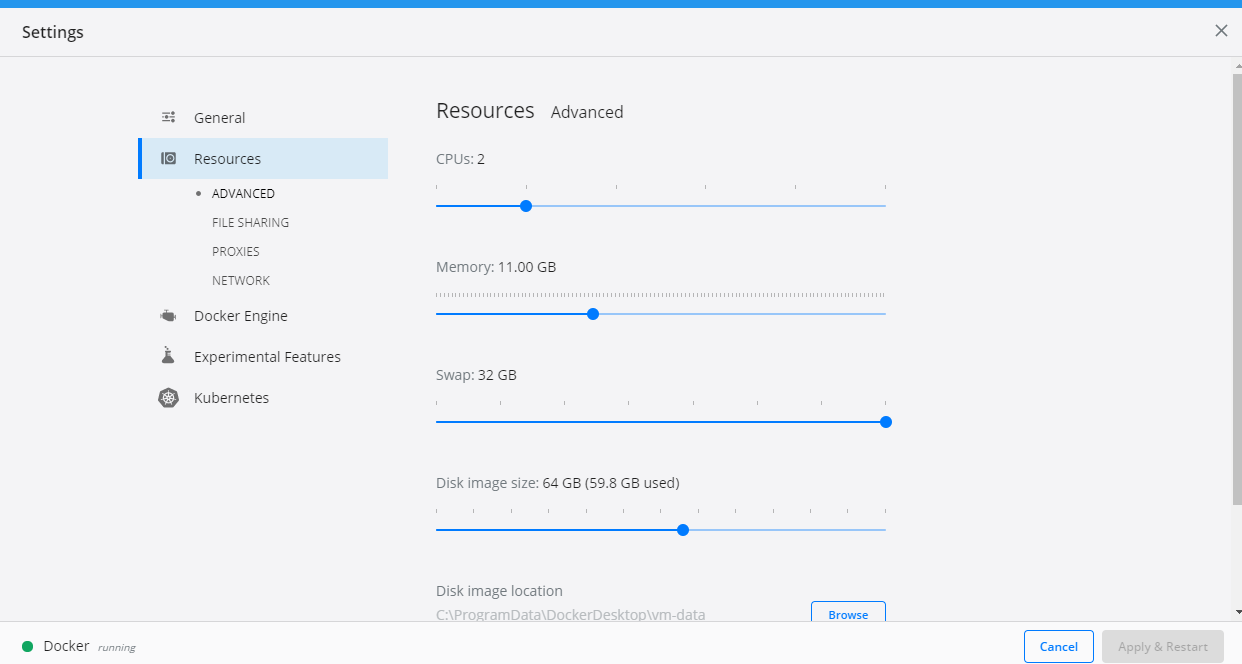
Installation process:

To run Astral, a Docker Desktop application is required. To download Docker Desktop, visit: <https://www.docker.com/products/docker-desktop>

After the installation process, you can launch Docker Desktop, and in the task bar, you should see Docker Desktop’s icon.



Right-click on it and navigate to **Settings**.



In the **Resources** tab, there are specifications for the computational power, that will be dedicated for Docker applications. You may manually customize, how much CPUs, Memory, Swap and Disk size you will give to Docker during runtime.

-------------------------------------------------------------------------------------------------------------------------

***HINT***:

If the Resources tab is not visible in the Docker panel, it is necessary to install Hyper-V service. The dialog window from the Docker Desktop should appear with the steps to install Hyper-V. In other case, visit:

<https://docs.microsoft.com/en-us/virtualization/hyper-v-on-windows/quick-start/enable-hyper-v>

-------------------------------------------------------------------------------------------------------------------------

For Astral to run properly, a significant amount of Virtual Memory is necessary. If possible, set Memory to at least 10GB. For Swap, set it to 32 GB.

Swap is a memory which is used when the application runs out of RAM. Because the timelapse data can be resource-consuming, it is advised to set it as high as possible.

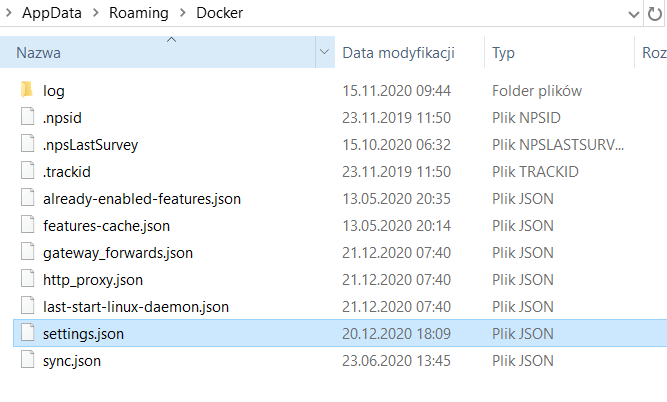
-------------------------------------------------------------------------------------------------------------------------

Hint: If you are unable to set Swap to higher values, open up Windows Run window and type:

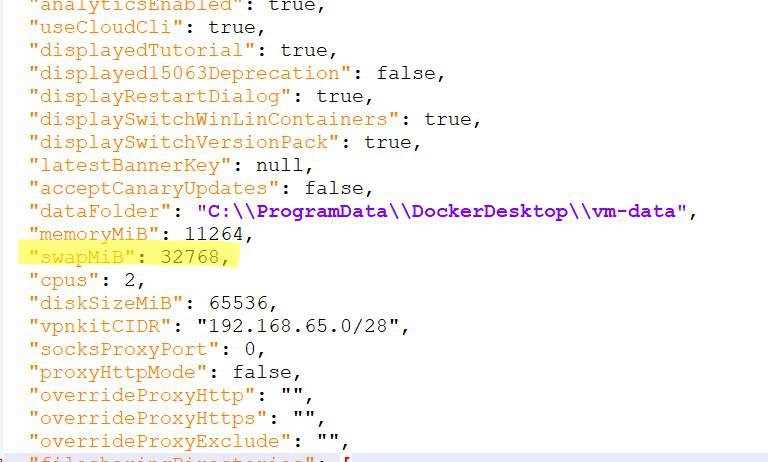
%appdata%

An explorer will show with an AppData folder. Then, navigate to the following file in the explorer.

AppData\Roaming\Docker\setting.json



And open it with your favorite editor. Then, to set the swap memory manually, find the swapMiB property:



And set it to a desired value.

-------------------------------------------------------------------------------------------------------------------------

Depending on assigned RAM memory amount, swap can be set empirically. For image series of 700Mb size and 8 GB of RAM, a good amount for swap is 32 GB, although the amount may be much smaller for some files. The amount of memory is really dependent on the number of events in the image series. This memory is released after application termination, so memory assignment is temporary.

**Warning**

Remember that swap is used when the program runs out of RAM memory. If the program will need to allocate resources in the swap, but there will not be enough swap memory, the task will crash. In case the step crashes, first thing to check is to increase swap memory and re-run the task.

# Installation and Running

## Prerequisites

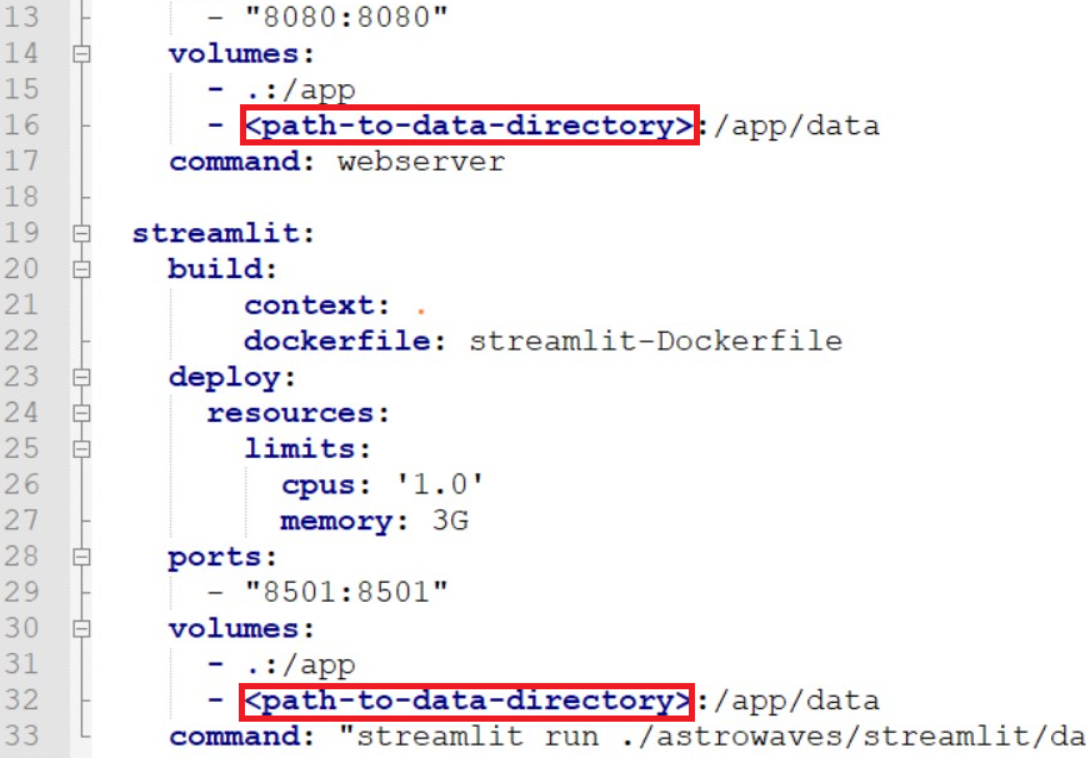
To run the Astral application, it is necessary to perform following steps.

First thing to do is to edit the **docker-compose.yml** file, which is located in the parent directory of the application. The file can be opened using any text editor.

The file specifies the internal configuration of the application. It consists of many values, which should not be modified, but some values are required to be customized.

One of such values is the path to the data directory, in which the image series files will be stored.

To run the Astral application, it is necessary to launch Powershell terminal. Following steps are necessary to launch the program:



Enter the path to the data directory, which contains the files to be processed by Astral. The path needs to be entered in two places (line 16 and line 32). **Do not remove the *:/app/data part* from the lines.**

This step is done once and does not need to be repeated, unless the data directory for image series is changed. In case the data directory is moved to another location, then it is necessary to modify the paths in the docker-compose.yml file.

1. Make sure Docker Desktop application is running, and enough RAM and swap memory is assigned.
2. Inside the Powershell command prompt, navigate to the Astral parent directory. To do this, type:

cd “<path-to-Astral-parent-directory”

For example:

cd “C:\Users\user\Documents \Astral”

If the command was executed successfully, a path specified from the command should be shown next to the cursor.

1. Astral application needs to be built – for that to happen, type:

Docker compose build



The application building process will start. The process can take up to several minutes, until the command prompt will be available for typing again.

1. Start the application by typing:

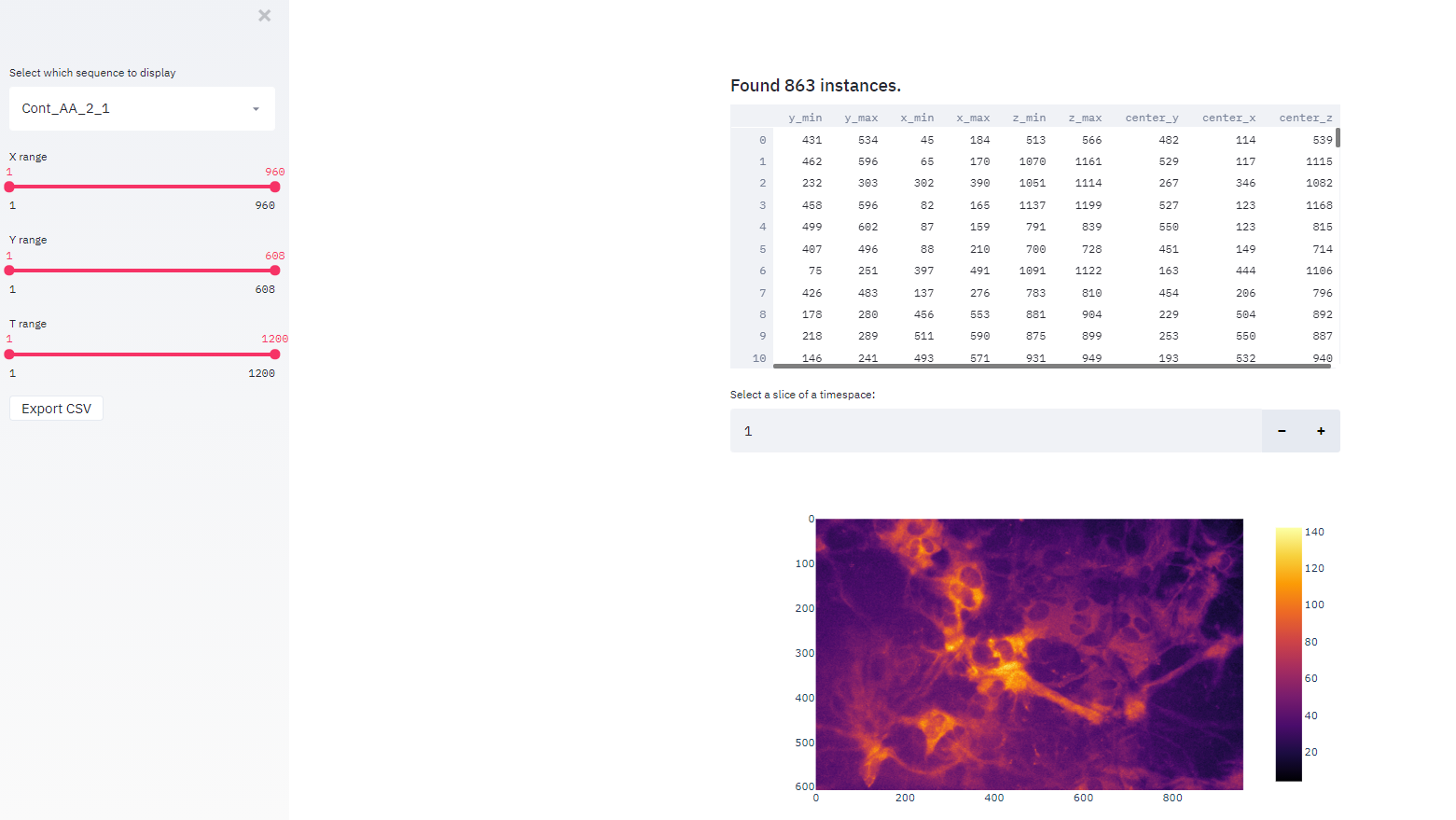
Docker compose up

Then, in the Powershell window, the application log files will start to show. The running application will have the similar output at the window:

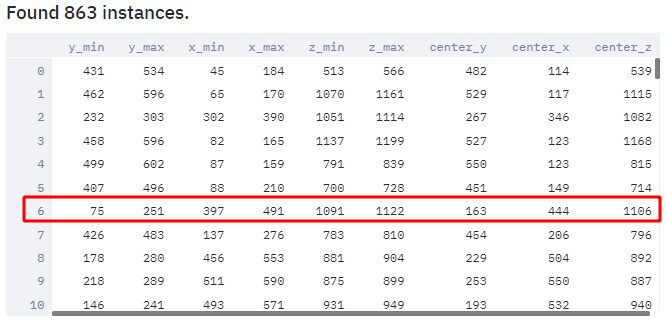
You can access the application on the URL address: http://localhost:8080/admin/ for Airflow application.

And on the URL address: <http://localhost:8501/> for the Streamlit application.

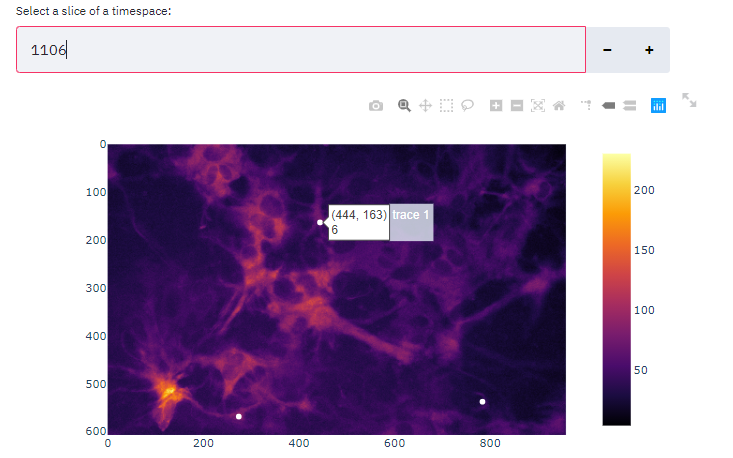
# Usage



Streamlit application enables the user to inspect the timelapse in terms of detected calcium waves. First, a list of all the calcium waves with their ids and coordinates is shown. To check the location of a given calcium wave in the timelapse, it is possible to navigate through the timelapse. For example:

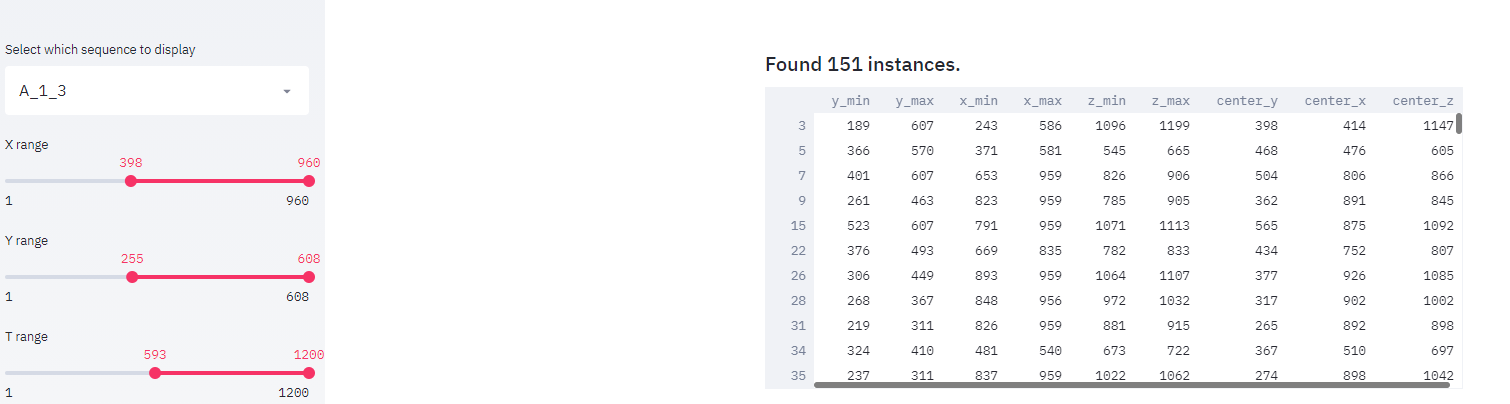


We would like to find the location in the series of the calcium wave with the id of 6. To do that:



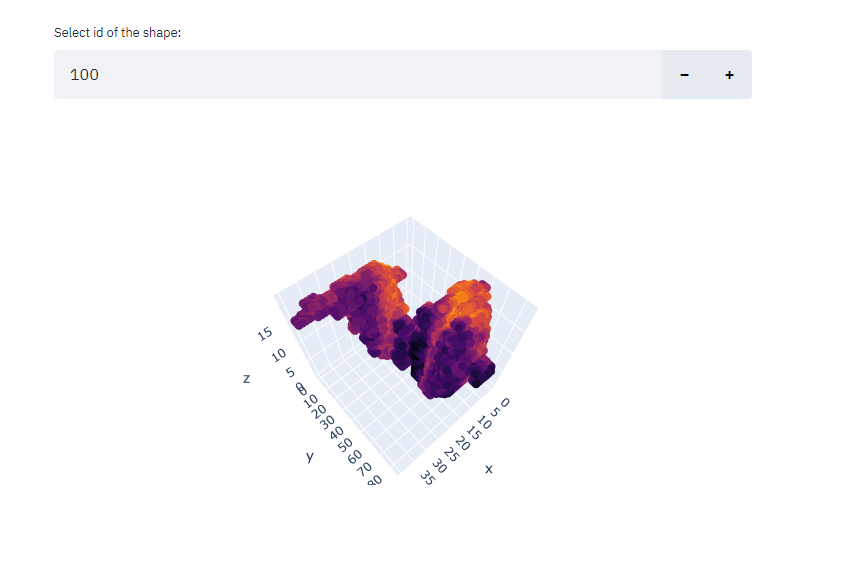
It is necessary to select a slice of a timespace of the center t coordinate of that calcium wave. Then, the calcium wave is marked with a white dot. In this case, two more calcium waves are also present on the slice.

Three sliders on the left can be used to filter the calcium wave list.



In this case, ranges in all the dimensions have been reduced. Only those calcium waves that are contained in this constrained subspace, are listed.

It is possible to view any calcium wave in 3D by selecting in by its id:



In this case, a calcium wave of id 100 is shown.

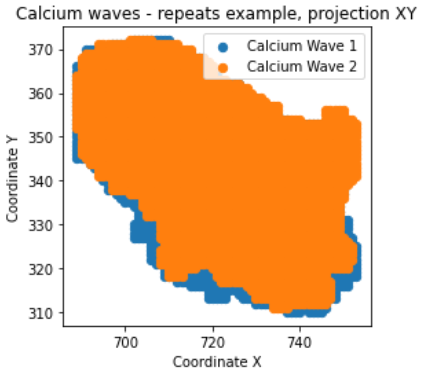
# Configuration

## Filename

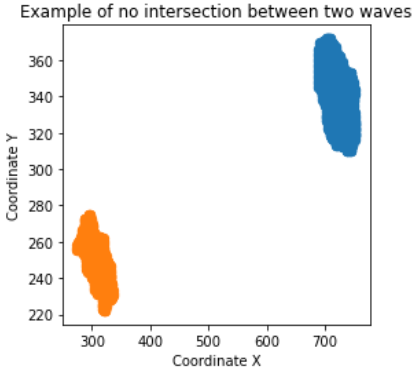
This parameter specifies the filename of the image series to be processed. If all the images inside the data directory should be analyzed, a special value for the parameter has been defined. To process all the files, “all” value should be entered.

## Intersect\_threshold

Defines, what is the threshold for for the Intersection Over Union between projections of the two calcium waves to consider those waves as repeats.



In the aboce example, the projections of the two calcium waves heavily overlap with each other. The value of intersection over union is high in this case, and if is greater than Intersect\_threshold, those will be treated as repeats.



In the above case, the Intersection Over Union is equal to 0, as there is no overlap between the waves. Unless the Intersect\_threshold is set to 0, those waves will not be treated as repeats.

## Standard\_deviation\_height\_range

This parameter specifies, how many consecutive pixels in the image should be taken into account during thresholding in y axis. The thresholding is the process of extracting signalling events from the image series. The parameter is used in ‘extract\_waves” task in “1\_extract\_waves” DAG.

## Standard\_deviation\_threshold

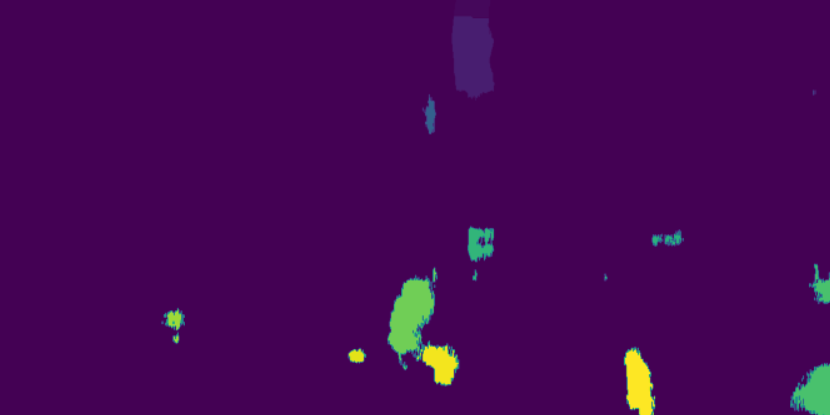
The threshold value for the standard deviation, above which a pixel is treated as calcium event. Pixels with values below the standard deviation threshold will be treated as background.

## Standard\_deviation\_width\_range

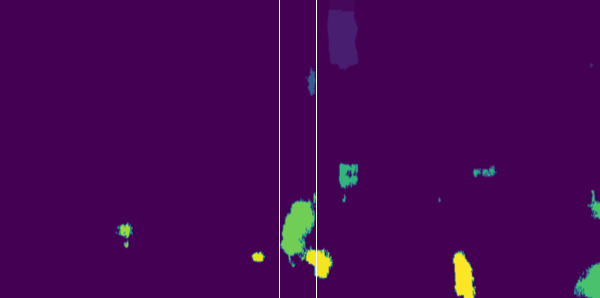
This parameter specifies, how many consecutive pixels in the image should be taken into account during thresholding in x axis. The thresholding is the process of extracting signalling events from the image series. The parameter is used in ‘extract\_waves” task in “1\_extract\_waves” DAG.

## Tolerance\_t

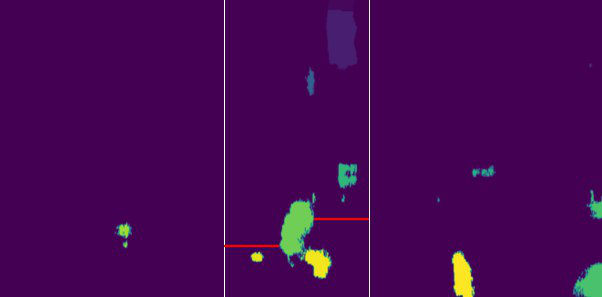
This parameter is used in “find\_neighbours” task in “3\_find\_neighbours” DAG. This parameters specifies, how far can calcium waves be separated in terms of t coordinate to be considered neighbours. For example:



For a green shape, a bounding box with respect to the t coordinate is drawn:



A bounding box is enlarged by tolerance\_z parameters from both sides:



The red lines’ length is specified by tolerance\_z parameter. Any calcium wave that is contained inside the defined region, is considered a neighbour.

## Tolerance\_xy

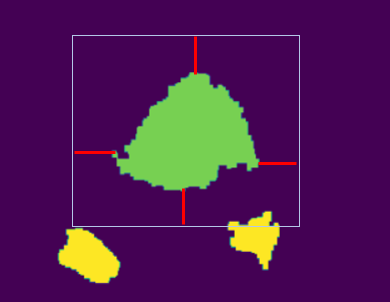
This parameter is used in “find\_neighbours” task in “3\_find\_neighbours” DAG. This parameter specifies, how far can calcium waves be separated in terms of x and y coordinate to be considered as neighbours. For example:



In the image, Astral has detected multiple calcium waves, marked with distinguishing colors. For a given calcium wave, a bounding box is drawn:



This bounding box is expanded in both x and y directions – the tolerance\_xy determines how far should the box be expanded:



The red lines indicate the size of the enlargement of the bounding box – each of those lines expanded the bounding box by tolerance\_xy parameter. In other words, the tolerance\_paramer has enlarged the bounding box by double the tolerance\_xy value in both x and y coordinates. Any shape that either crosses the region defined by the bounding box is considered a neighbour. In the case above, one of the waves is a neighbour, as it crosses the boundaries, whereas the other one is not considered a neighbour.

## Use\_watershed

This parameter is used in “Create\_masks” task in “1\_extract\_waves” DAG. If the parameter is set to 1, the task additionally uses Watershed algorithm for splitting adjacent waves apart. If the parameter is set to 0, the task does not perform this additional step. Below you may find an example of a 2D scenario for a Watershed algorithm:

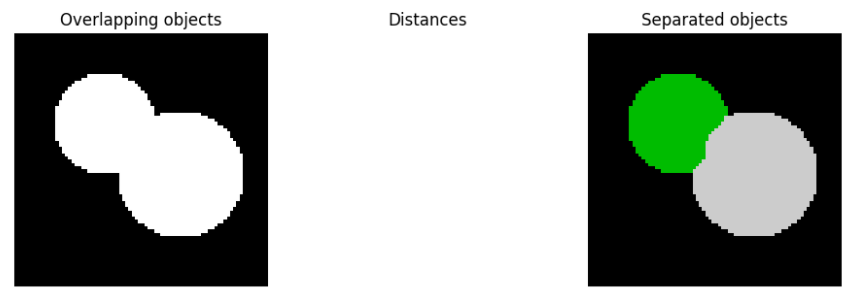


Figure 1. Splitting of overlapping objects. Source: <https://scikit-image.org/docs/dev/auto_examples/segmentation/plot_watershed.html>

The algorithm works in 3D scenario. Basing on intensity of the pixels forming a calcium waves, it determines whether a case contains multiple waves joined together.

## Volume\_threshold

Volume threshold is used in “label\_waves” task in “2\_segment\_waves” DAG. It drops calcium waves, which consist of less pixels than volume\_threshold. If the sum of all wave’s pixels (across all timesteps) is less than the threshold, it is disregarded from further analysis.