# User manual sdt-pics

## Installation instructions

### Create a Docker image and Docker container

1. Make sure you have Docker installed (`https://docs.docker.com/engine/install/`)
2. Clone the Bitbucket directory

```
git clone https://wimthiels@bitbucket.org/pgmsembryogenesis/sdt-pics.git
```

3. Place yourself inside the extracted folder run the following bash command:

```
./docker_build.sh
```

   This will create two docker images (sdt and mpacts) and a docker container for each (sdt_container and mpacts_container). Check if the 2 containers are up and running with 'docker ps'
   **Important Note**: A stable internet connection is required. If a time-out/download error occurs, simply relaunch the command.

### Run your first test

To check if everything works as it should, run a first test:

```
./sdt-pics.sh pics_test
```

If all worked fine you should see output in folder `DATA/OUTPUT/pics_test`. The subfolder `021_mpacts_pics_select` contains cell meshes in both vtp and stl-format. The stl-files are the plain 3D segmentations, while the vtp cells contain extra data arrays pertaining to the mechanical simulation. You can vizualise these cells using mesh visualization software, such as Paraview (`https://www.paraview.org/download/`)

## Using sdt-pics

### Running a testcase (basic use)

The following section will explain the basic principles and will show you how to apply sdt-pics on your own microscopy data.

sdt-pics currently has no GUI and must is run by executing the bash script **sdt-pics.sh**. The state of execution, meaning the program flow, path names and parameters, is defined in a separate XML-file (**PARAMS.xml**). This file also serves to handle default values and offers a description for each of the parameters. One could modify **PARAMS.xml** directly and run test cases that way, but the preferred way is to instead edit the **testinfra_docker.xls** file, which acts as a state modifier. All the fields mentioned here will be automatically changed in **PARAMS.xml** before execution. The advantage of working with this excel file is that you can define as many test cases as you want (one test case = one tab), while only listing the fields that you want modified, which is more concise and clear.

**Set up** the testcase:

- Add a tab in **testinfra_docker.xls**. The tab name equals your test case name. In this excel you will find a template tab (Figure 1A) that you can copy and modify (Figure 1 B).
- add a line in **testinfra_execution.txt** for your testcase:

```
EXE-;<your_testcase_name>;<group_label>;_docker
```

- Extra parameters can be modified by adding rows (make sure the value of the first column points to the xml-key used in PARAMS.xml).

When the testcase is set up, you can **run** it as follows:

```
./sdt-pics.sh my_testcase
```

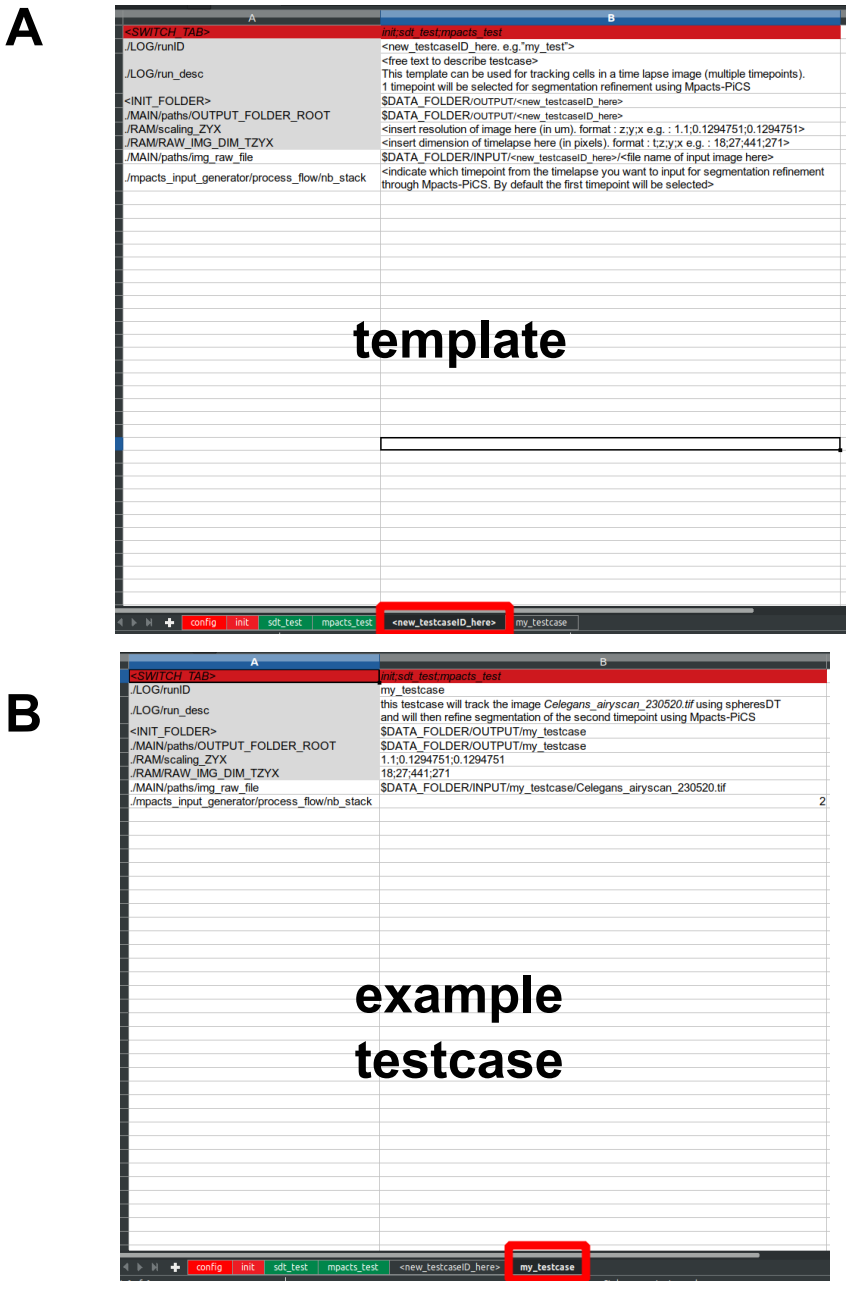


Figure 1: Setting up a testcase via the testinfra_docker.xls file

## advanced use

### Running multiple testcases

You can run a set of testcases by giving them a group label in testinfra_execution.txt:

```
EXE-;<your_testcase_name>;<group_label>;_docker
```

and then run them in group (in series) by

```
./sdt-pics.sh -g <group_label>
```

Or, for more ad-hoc grouping, you can mark them by giving them the 'EXEC' label:

```
EXEC;<your_testcase_name>;<group_label>;_docker
```

and run all testcases with the EXEC label as follows (no arguments):

```
./sdt-pics.sh
```

## Parameters

As mentioned before, PARAMS.xml groups all parameters in one file, and also offers a short description. Below is a list of the most important parameters. The ones labeled 'basic' should be considered first, while the 'advanced' offer more fine-grained control and are probably most relevant in the initial stages when calibrating the method to the specifics of the data (type of microscope, fluorescent marker used, S/N level ...).

| | | |
|---|---|---|
| SEED_THR_DIVIDE_FACTOR | preprocessing | advanced |
| The seed threshold determined by automatic thresholding (Otsu) will be divided by this factor. Use in case not enough/too much seed signal is picked up. e.g. '2' will lower the Otsu's seed threshold by half. | | |
| MEMBRANE_ACCEPTANCE_LEVEL | preprocessing | basic |
| This will determine the bottom threshold and is used a fraction of the otsu threshold for that slice(Otsu itself will provide the seed threshold). higher values will drive down the bottom threshold. If set to 1 then seeds will not be extended. If set lower than 1 threshold will be higher than seed threshold, so seeds will be reduced or even erased | | |
| NO_SIGNAL_THRESH | preprocessing | advanced |
| If the seed threshold falls below this level, the slice is assumed to have no signal, only noise | | |
| PEAK_THRESH_REL | preprocessing | advanced |
| Affects the detection of membranes parallel to the focal plane. Minimum intensity of peaks, calculated as max(image) * peak_thresh_rel. Make smaller to increase nb of peaks | | |
| FRAGMENTATION_LEVEL | spheresDT | basic |
| This influences the overlap threshold used in cluster and cell creation. Increasing it, will increase the overlap required for 2 spheres or clusters to group together, and therefore boosting cell division. The value lies between 0 and 1. It reflects a percentage of the radius of the sphere (the seed sphere in case of cluster) involved. 0 = every overlap will cause clusters to be fused, 0.75 = fuse if you can move the smallest cluster to the other if you shrink its radius to 75% its original size 1 = only fuse if the overlap hole (=maximum DT value in the overlap zone) is so big the smallest cluster can move to the other cluster without shrinking it. This is considered a user parameter | | |
| MIN_SPHERE_RADIUS | spheresDT | basic |
| Threshold for sphere creation. The lower, the higher the resolution and computation time | | |
| MIN_CELL_RADIUS | spheresDT | basic |
| Threshold for cell creation (used in first stack and cell division). This is defined as the radius the volume of a sphere that can harbor all pixels of the clusters. This is considered a user parameter | | |
| | | |

Table 1: User parameters