transphire Documentation

Release 0.0.1

Markus Stabrin

Jul 12, 2023

Contents:

TranSPHIRE is an automated pre-processing tool designed for on-the-fly processing during data aquisition. It is an open source project published under the GPLv3 license and the code is available on GitHub.

TranSPHIRE coveres the inital steps of the Single Particle Analysis pipeline.

CHAPTER 1

Pre-processing

- **Motion correction (Outputs can be used to run particle polishing in Relion)**
  - MotionCor2 - (Not free for commercial use, a license needs to be bought separately)
  - Unblur
- **CTF estimation**
  - SPHIRE CTER
  - CTFFIND4
  - GCtf
- **Particle picking**
  - SPHIRE crYOLO - (Not free for commercial use, a license needs to be bought separately)
- **Particle extraction**
  - SPHIRE WINDOW

CHAPTER 2

Processing

- **2D classification**
  - SPHIRE GPU ISAC
- **2D class selection**
  - SPHIRE Cinderella
- **3D initial model estimation**
  - SPHIRE RVIPER
- **3D refinement**
  - SPHIRE MERIDIEN

CHAPTER 3

Feedback loop

Additionally, TranSPHIRE implements a new `Feedback loop` that automatically re-trains and therefore adapts the model used for particle picking to the data set at hand.

1. Particle picking
2. Particle extraction
3. Wait for a number of extracted particles to accumulate
4. 2D classification
5. 2D class selection
6. Class member extraction
7. Re-training of the picking model
8. `1.` with the re-trained model

CHAPTER 4

Hardware recommendations

Hardware recommendations can be found at the *Hardware recommendations* page.

CHAPTER 5

Installation

The installation instructions can be found at the *Installation* page.

CHAPTER 6

Tutorial

The tutorial can be found at the *Tutorial* page.

CHAPTER 7

Contribute

If you want to contribute to the TranSPHIRE project, please checkout the *How to contribute* page.

CHAPTER 8

License

TranSPHIRE is an open source project published under the GPLv3 license.

CHAPTER 9

Code availability

The source code is available on GitHub.

9.1 Welcome to TranSPHIRE's documentation!

TranSPHIRE is an automated pre-processing tool designed for on-the-fly processing during data aquisition. It is an open source project published under the GPLv3 license and the code is available on GitHub.

TranSPHIRE coveres the inital steps of the Single Particle Analysis pipeline.

9.1.1 Pre-processing

- **Motion correction (Outputs can be used to run particle polishing in Relion)**
  - MotionCor2 - (Not free for commercial use, a license needs to be bought separately)
  - Unblur
- **CTF estimation**
  - SPHIRE CTER
  - CTFFIND4
  - GCtf
- **Particle picking**
  - SPHIRE crYOLO - (Not free for commercial use, a license needs to be bought separately)
- **Particle extraction**
  - SPHIRE WINDOW

9.1.2 Processing

- **2D classification**
  - SPHIRE GPU ISAC
- **2D class selection**
  - SPHIRE Cinderella
- **3D initial model estimation**
  - SPHIRE RVIPER
- **3D refinement**
  - SPHIRE MERIDIEN

9.1.3 Feedback loop

Additionally, TranSPHIRE implements a new `Feedback loop` that automatically re-trains and therefore adapts the model used for particle picking to the data set at hand.

1. Particle picking
2. Particle extraction
3. Wait for a number of extracted particles to accumulate
4. 2D classification
5. 2D class selection
6. Class member extraction
7. Re-training of the picking model
8. `1.` with the re-trained model

9.1.4 Hardware recommendations

Hardware recommendations can be found at the *Hardware recommendations* page.

9.1.5 Installation

The installation instructions can be found at the *Installation* page.

9.1.6 Tutorial

The tutorial can be found at the *Tutorial* page.

9.1.7 Contribute

If you want to contribute to the TranSPHIRE project, please checkout the *How to contribute* page.

9.1.8 License

TranSPHIRE is an open source project published under the GPLv3 license.

9.1.9 Code availability

The source code is available on GitHub.

9.2 Hardware recommendations

We have the following GPU machines in use:

- Screening (Talos Arctica - Falcon3 linear)
  - Intel(R) Xeon(R) CPU E5-2630 v4 @ 2.20GHz (10 cores / 20 cores hyperthreading)
  - 64 GB RAM
  - GeForce GTX 1080
  - 30 TB raid hard drives
- Data acquisition (Titan Krios - K3 super resolution)
  - 2x Intel(R) Xeon(R) Gold 6128 CPU @ 3.40GHz (6 cores / 12 cores hyperthreading)
  - 192 GB RAM
  - 3x GeForce GTX 1080 Ti
  - 55 TB raid hard drives
  - 2 TB scratch SSD

9.3 Changelog

9.3.1 Version 1.5

Version latest (Current master branch, might be unstable)

- Fix an telegram issue when the messages do not contain proper key entries
- Reduce the margin around the logfile box
- Fixed a problem with with GPU SPLIT parameter when unchecked global flag in the GUI GPU parameter
- Fixed a crash with connection refused SMTP error
- Expose –batch_size option in cryolo train
- Do not do checksum calculations if the input file to copy is a directory
- Proper restart of Motion with CTF Movie mode
- Try to copy 5 times before deciding that a file is not able to copy

Version 1.5.13

- Set project pattern as default for the Project Name
- Fix problem of mistyped e2proc2d.py option unstacking that lead to crashes

Version 1.5.11

- Improved filament mode for ISAC
- Fixed retraining
- Fix latency problems with arriving jpg files
- Fix visualization problems with Sphire 1.4
- Minor bug fixes

Version 1.5.0

- First public release
- Automated data processing
- TranSPHIRE feedback loop

9.4 Installation

Welcome to the installation page of TranSPHIRE. The installation can be divided into three parts.

1. *Installation of TranSPHIRE*
2. *Installation of the dependencies*
3. *Basic TranSPHIRE setup*

The TranSPHIRE version changelog can be found here:

1. *Changelog*

9.4.1 Installation of TranSPHIRE

In order to install TranSPHIRE it is highly recommended to setup an Anaconda / Miniconda environment. The same installation can be used to install crYOLO and other dependencies later. In case you want to learn more about conda environments: Manage environments.

After successful installation, the conda command should be available. To make things easier for copy and pasting, the proposed commands utilize bash variables. The installation does usually not take longer than a few minutes.

0. Specify Necessary variables

```
>>> TRANSPHIRE_ENV_NAME=transphire
```

1. Create a conda environment

```
>>> conda create -n ${TRANSPHIRE_ENV_NAME} python=3.6 pyqt=5
```

2. Activate the environment

```
>>> conda activate ${TRANSPHIRE_ENV_NAME}
```

Note: For older versions of conda it might be:

```
>>> source activate ${TRANSPHIRE_ENV_NAME}
```

3. Install TranSPHIRE

```
>>> pip install transphire
```

4. Test the installation

```
>>> transphire --version
```

9.4.2 Installation of the dependencies

TranSPHIRE is a wrapper for already existing software packages available. Therefore, it is necessary to install the dependencies separately. In the future, we will work on a dependency installer tool.

Utilities and packages

- SPHIRE version >= 1.4 / SPHIRE version >= 2.31
- IMOD version >= 4.9
- CHIMERAX version >= 1.0 - (Not free for commercial use, a license needs to be bought separately)

Note: The SPHIRE installation automatically installs all necessary tools for

- Particle extraction
- 3D initial model estimation
- 3D refinement

and installs utility programs from the EMAN2 package.

Note: The SPHIRE installation needs one additional command to function properly:

```
>>> bash ${SPHIRE_INSTALL_DIR}/utils/replace_shebang.sh
```

This script will replace the shebang line of the SPHIRE executables to avoid collision with other python interpreters in your PATH.

Note: for the IMOD installation, you need to make sure that the IMOD source file is sourced in order to run properly.

Motion correction

- MotionCor2 version >= 1.0.0 - (Not free for commercial use, a license needs to be bought separately)
- Unblur cisTEM version >= 1.0.0-beta

CTF estimation

- SPHIRE CTER
- CTFFIND4 version >= 4.1.8
- GCtf version >= 1.06

Note: GCtf version 1.18 is sometimes behaving different than expected. Use with caution.

Particle picking

- SPHIRE crYOLO version >= 1.0.4 - (Not free for commercial use, a license needs to be bought separately)

Note: crYOLO cannot be installed within the TranSPHIRE anaconda environment. Fortunately, this is not a problem, due to the total independence of anaconda environments. After following the installation instructions of crYOLO and installed it in a seperate environment just deactivate the crYOLO environment and activate the *TranSPHIRE* environment again. Just provide the link to the executable crYOLO file in the TranSPHIRE GUI. Those are usually located in *${_CONDA_ROOT}/envs/CRYOLO_ENV_NAME/bin*. The information how and in which environment to execute the respective executables is provided in the header. Alternatively, the directory path *${_CONDA_ROOT}/envs/CRYOLO_ENV_NAME/bin* can be added to the PATH variable.

2D classification

- SPHIRE GPU ISAC version >= 2.3.1

2D class selection

- SPHIRE Cinderella version >= 0.3.1

Note: Cinderella cannot be installed within the TranSPHIRE anaconda environment. Fortunately, this is not a problem, due to the total independence of anaconda environments. After following the installation instructions of Cinderella and installed it in a seperate environment just deactivate the Cinderella environment and activate the *TranSPHIRE* environment again. Just provide the link to the executable Cinderella file in the TranSPHIRE GUI. Those are usually located in *${_CONDA_ROOT}/envs/CINDERELLA_ENV_NAME/bin*. The information how and in which environment to execute the respective executables is provided in the header. Alternatively, the directory path *${_CONDA_ROOT}/envs/CINDERELLA_ENV_NAME/bin* can be added to the PATH variable.

9.4.3 Presentations

2020 10 29 - SBGrid Consortium YouTube

9.4.4 Basic TranSPHIRE setup

The following instructions are suppose to run only once at the very beginning after installation of TranSPHIRE.

Command line arguments

TranSPHIRE offers some command line arguments that control its basic behaviour. Most of the command line arguments can additionally be controlled by environmental variables. This is especially useful if TranSPHIRE is used in an `environmental module` environment.

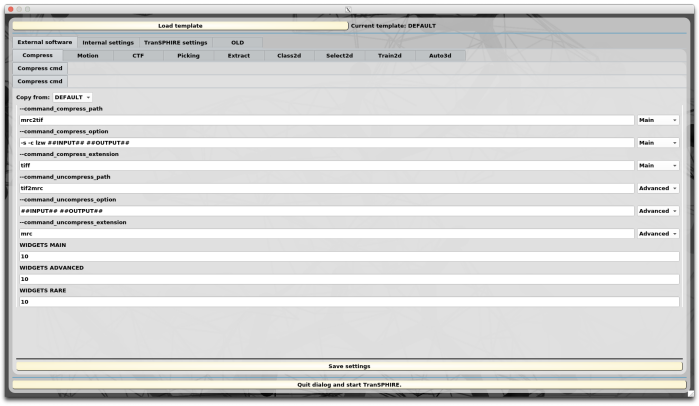
Commmandline options / Environmental variable	Description	Default value
–root_directory TRANSPHIRE_ROOT_DIRECTORY	TranSPHIRE root directory. This is the directory where TranSPHIRE will be started. Every provided relative directory and file path will be respective to this directory.	The users home directory.
–settings_directory TRANSPHIRE_SETTINGS_DIRECTORY	TranSPHIRE settings directory. This is the directory where the settings and templates are stored.	transphire_settings in the root_directory
–mount_directory TRANSPHIRE_MOUNT_DIRECTORY	TranSPHIRE mount directory. In case users need to mount pre-defined mount points themselves, those are located in this directory.	transphire_mounts in the root_directory
–font TRANSPHIRE_FONT_SIZE	Font size to use within the TranSPHIRE GUI. Most widgets are scaled accordingly.	5 or read from settings if not provided.
–adjust_width TRANSPHIRE_ADJUST_WIDTH	Scaling factor for the widget width. >1 will make the widgets larger; <1 will make the widgets smaller.	1 or read from settings if not provided.
–adjust_height TRANSPHIRE_ADJUST_HEIGHT	Scaling factor for the widget height. >1 will make the widgets larger; <1 will make the widgets smaller.	1 or read from settings if not provided.
–n_feedbacks TRANSPHIRE_N_FEEDBACKS	Maximum number of allowed feedbacks.	10
–edit_settings	Open the “Default settings” dialog.	–
–version	Show version information.	–
–kill	Kill open, dead or stalling TranSPHIRE runs.	–

Basic setup and templates

It is possible to setup default settings and templates for the TranSPHIRE pipeline. To enter the setup area type:

```
>>> transphire --edit_settings
```

And the GUI will open:



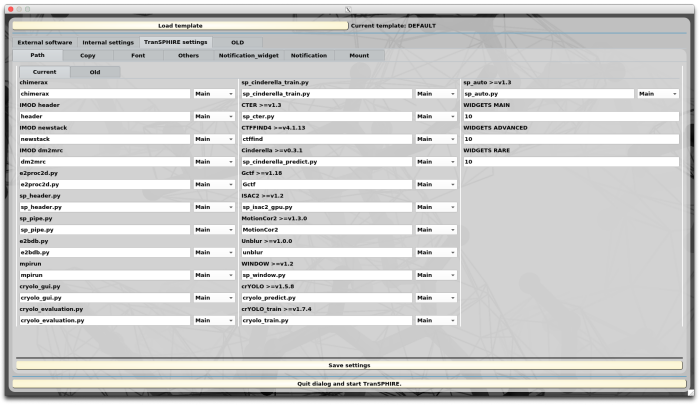
2020-09-03 at 10.34.03.png

The first level Tab bar groups the different settings.

- Settings affected by templates
  - External software: Contains all settings related for the software packages used for processing.
  - Internal settings: Settings related to the TranSPHIRE pipeline itself.
- Settings shared throughout templates
  - TranSPHIRE settings: Settings that should be available to every template.

Program paths

Click on **TranSPHIRE settings -> Path -> Current**.

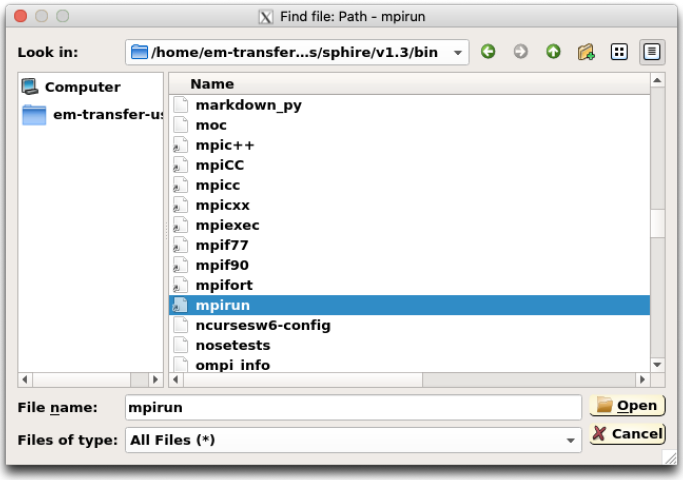


2020-09-03 at 11.01.54.png

Provide the full PATH to the executables. You can press **Ctrl + Return** while editing to open a *File Open Dialog*. If you are not sure where the files are located, but you can execute them in the terminal type:

```
>>> which desired_command
```

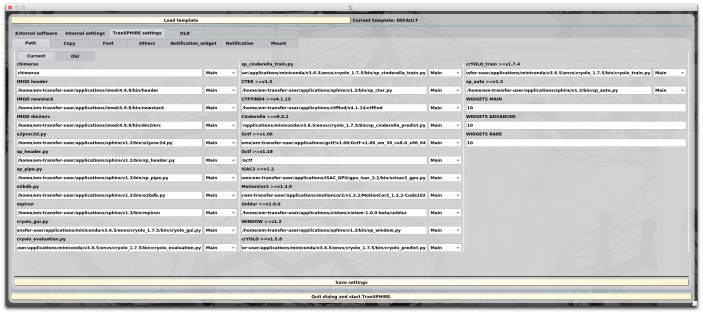
Otherwise contact your system administrator.



2020-09-03 at 11.57.13.png

Note: Please provide the mpirun command that is shipped with the SPHIRE installation.

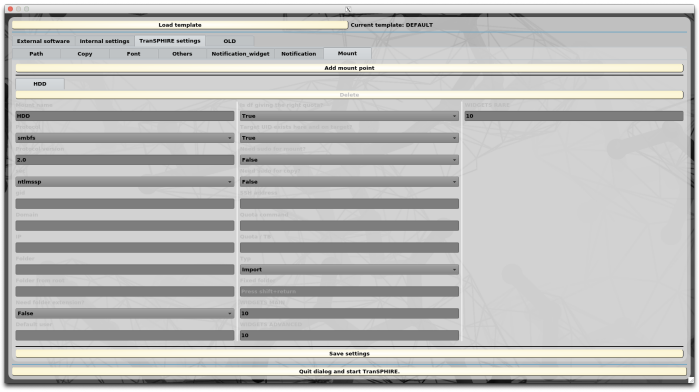
After the information has been provided, click the **Save settings** button.



2020-09-03 at 12.14.36.png

Mount points

Next provide the mount point information to tell TranSPHIRE where possible data is located or where to copy created data to. Click on **TranSPHIRE settings -> Mount**. By default there is a mount point for external hard drives present, but additional mount points need to be added in order to function properly. To mount external Machines, Linux mount.mount_protocol executables are used.



2020-09-03 at 12.47.46.png

There are two possible ways to deal with mount points: *Fixed folder mount points* and *On demand mount points*.

Fixed folder mount points

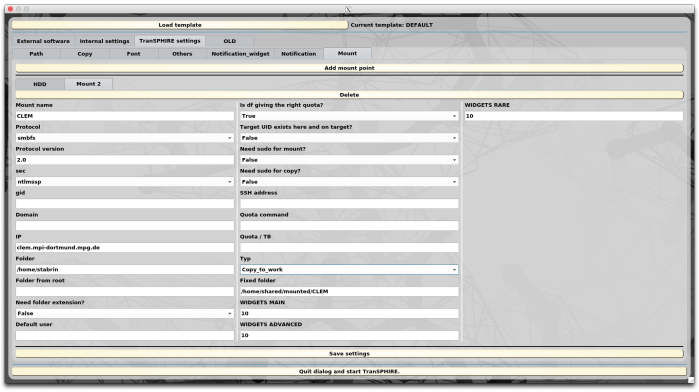
Click on the **Add mount point** button and a new mount point entry `Mount 2` appears. You need to provide the following entries:

- **Mount name**
  - Name of the mount point within the TranSPHIRE GUI.
- **IP**
  - IP adress of the remote device. This is necessary to assure the correct execution of the Auto3D command via SSH.
- **Folder**
  - Mount entry folder name of the remote device.
- **Typ**
  - Choose if the mount point is used to import the data or if it is a destination for either processing or backup.
- **Fixed folder**
  - Folder of the fixed mount point on the local device.

Note: Our cluster is called `CLEM` and is mounted on the local device at the location `/home/shared/mounted/CLEM`. The IP is `clem.mpi-dortmund.mpg.de` and the mounted folder is `/home/stabrin`. Because it is a cluster for data processing, the Typ is `Copy_to_work`. The Fixed folder location is `/home/shared/mounted/CLEM`.

Therefore our configuration is:

- **Mount name**: CLEM
- **IP**: clem.mpi-dortmund.mpg.de
- **Folder**: /home/stabrin
- **Typ**: Copy_to_work
- **Fixed folder**: /home/shared/mounted/CLEM



2020-09-03 at 15.20.49.png

On demand mount points

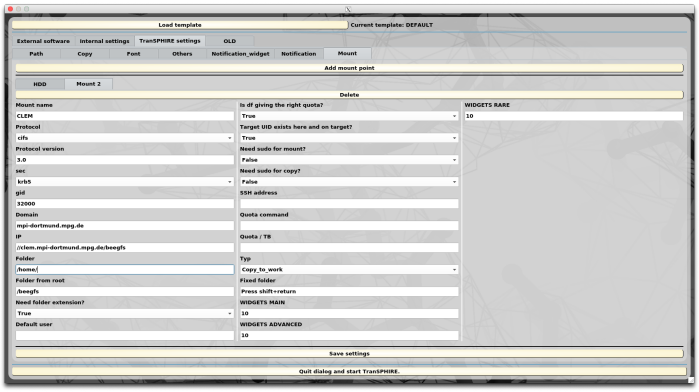
Click on the **Add mount point** button and a new mount point entry `Mount 2` appears. To fill out the respective entries, you should talk to your system administrator. You need to provide the following entries:

- **Mount name**
  - Name of the mount point within the TranSPHIRE GUI.
- **Protocol**
  - Mount protocol.
- **Protocol version**
  - The version of the protocol.
- **sec**
  - The security protocol used for the mount point. If your `sec` value is `krb5`, the cruid option is automatically set to the user. If you need a different behaviour, please contact markus.stabrin@mpi-dortmund.mpg.de.
- **gid**
  - The mount group.
- **Domain**
  - Domain of the mount point.
- **IP**
  - IP adress of the remote device. This is necessary to assure the correct execution of the Auto3D command via SSH.
- **Folder**
  - Mount entry folder name of the remote device.
- **Folder from root**
  - Path to the folder specified in `Folder` from the root directory of the remote device.
- **Need folder extension?**
  - Set to True, to allow dynamic point entries. See the note for an example.
- **Default user**
  - Default user to fill in for mounting. This way only the password needs to be provided. Useful for computers where the mount user does not change.
- **Is df giving the right quota?**
  - The Linux command `df` provides information about the disc occupancy of mount points. However, for file systems that use a quota management this value is usually not correct. If in doubt, leave the settings to `True`.
- **Target UID exists here and on target?**
  - For domain user managed computers. If set to `True`, the provided user/password combination will be used to run an `ls` command on the local machine to provide a sanity check.
- **Need sudo for mount?**
  - **WARNING**: Providing root passwords is not ideal. We would recommended to use `cifs` mount points and allow for password-less sudo rights for `mount.cifs`.

- **Need sudo for copy?**
  - **WARNING**: Use this with caution, a shared account for the TranSPHIRE runs is not recommended. If you use a shared "Transfer" account for the dedicated TranSPHIRE machine, the root password needs to be provided at the beginning of the session to allow for copy of the data to the mount points.
- **SSH address**
  - SSH address used to calculate the quota if `Is df giving the right quota` is set to False.
- **Quota command**
  - Command to calculate the quota on the remote device.
- **Quota / TB**
  - **Deprecated** will be removed in the next versions.
- **Typ**
  - Choose if the mount point is used to import the data or if it is a destination for either processing or backup.
- **Fixed folder**
  - Folder of the fixed mount point on the local device.

Note: The current user is `stabrin` and the authentification mechanism works with a `kerberos ticket`. Our cluster is called `CLEM`. The mount protocol is `cifs` and the version is `3.0`. The sec protocol is `krb5`. `stabrin` is a member of the group `32000` ad the domain is `mpi-dortmund.mpg.de`. The mount IP entry point is `//clem.mpi-dortmund.mpg.de/beegfs` and the mount folder is `/home/`. The path from root to the mount folder is `/beegfs` resulting in `/beegfs/home/` on the remote device. Since every user has its own separate home directory on `CLEM` and we want to allow for dynamic mounting, we have `Need folder extension?` set to `True`. This way we can provide `stabrin` as the folder extension during mounting to mount `/beegfs/home/stabrin`. Additionally, the `Target UID exists here and on target` and we change the setting to `False` as well as the `Typ` to `Copy_to_work`.

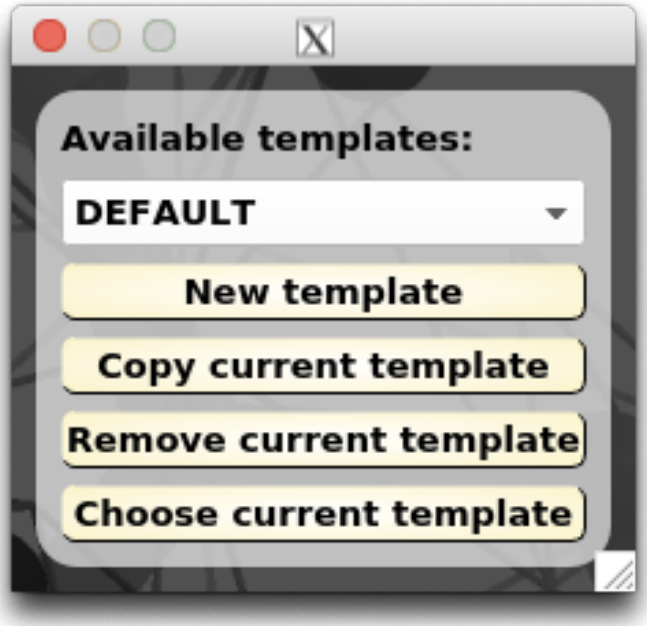
Therefore our configuration is:



2020-09-03 at 16.15.17.png

Create templates

In order to create setting templates you need to click the **Load template** button.



2020-09-03 at 16.39.59.png

- Drop-down widget: You can choose the template here.
- New template: You can create a new and empty template entry.
- Copy current template: Create a new template, but the settings are identical to the template chosen in the drop-down widget.
- Remove current template: Remove the currently selected template.
- Choose current template: Choose the current template in order to change its settings.

Click on **New template** and provide a name like `Tutorial_template` for your new template and set **Choose current template**. The text next to the **Load template** button indicates that the correct template is active.

2020-09-03 at 16.53.52.png 

Now adjust the settings to match the needs for your facility. This helps especially beginner users to avoid making errors during setup.

9.5 Tutorial

9.5.1 Presentations

2020 10 29 - SBGrid Consortium YouTube

9.5.2 Basic TranSPHIRE setup

The following instructions are suppose to run only once at the very beginning after installation of TranSPHIRE.

Command line arguments

TranSPHIRE offers some command line arguments that control its basic behaviour. Most of the command line arguments can additionally be controlled by environmental variables. This is especially useful if TranSPHIRE is used in an `environmental module` environment.

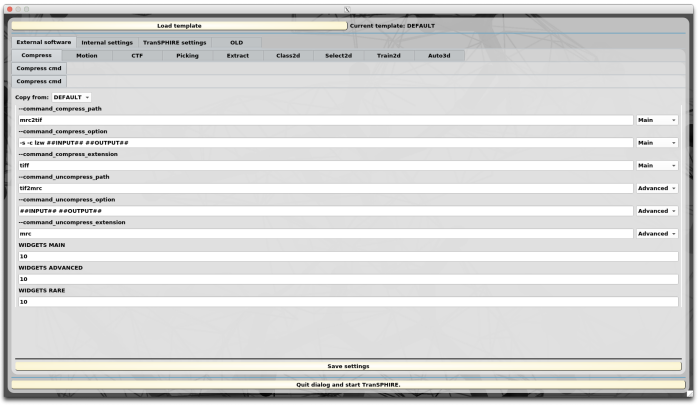
Commmandline options / Environmental variable	Description	Default value
–root_directory TRANSPHIRE_ROOT_DIRECTORY	TranSPHIRE root directory. This is the directory where TranSPHIRE will be started. Every provided relative directory and file path will be respective to this directory.	The users home directory.
–settings_directory TRANSPHIRE_SETTINGS_DIRECTORY	TranSPHIRE settings directory. This is the directory where the settings and templates are stored.	transphire_settings in the root_directory
–mount_directory TRANSPHIRE_MOUNT_DIRECTORY	TranSPHIRE mount directory. In case users need to mount pre-defined mount points themselves, those are located in this directory.	transphire_mounts in the root_directory
–font TRANSPHIRE_FONT_SIZE	Font size to use within the TranSPHIRE GUI. Most widgets are scaled accordingly.	5 or read from settings if not provided.
–adjust_width TRANSPHIRE_ADJUST_WIDTH	Scaling factor for the widget width. >1 will make the widgets larger; <1 will make the widgets smaller.	1 or read from settings if not provided.
–adjust_height TRANSPHIRE_ADJUST_HEIGHT	Scaling factor for the widget height. >1 will make the widgets larger; <1 will make the widgets smaller.	1 or read from settings if not provided.
–n_feedbacks TRANSPHIRE_N_FEEDBACKS	Maximum number of allowed feedbacks.	10
–edit_settings	Open the “Default settings” dialog.	–
–version	Show version information.	–
–kill	Kill open, dead or stalling TranSPHIRE runs.	–

Basic setup and templates

It is possible to setup default settings and templates for the TranSPHIRE pipeline. To enter the setup area type:

```
>>> transphire --edit_settings
```

And the GUI will open:



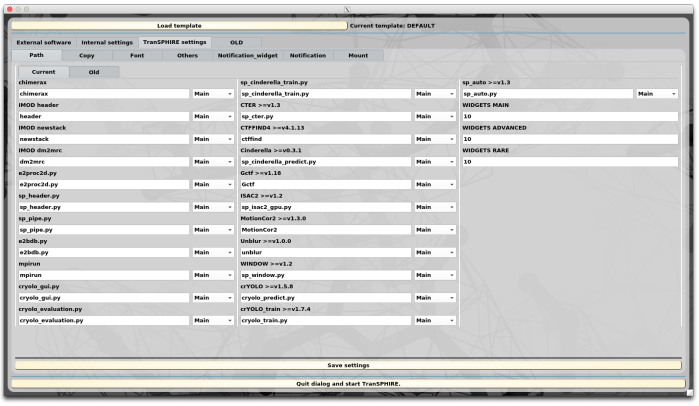
2020-09-03 at 10.34.031.png

The first level Tab bar groups the different settings.

- Settings affected by templates
  - External software: Contains all settings related for the software packages used for processing.
  - Internal settings: Settings related to the TranSPHIRE pipeline itself.
- Settings shared throughout templates
  - TranSPHIRE settings: Settings that should be available to every template.

Program paths

Click on **TranSPHIRE settings -> Path -> Current**.

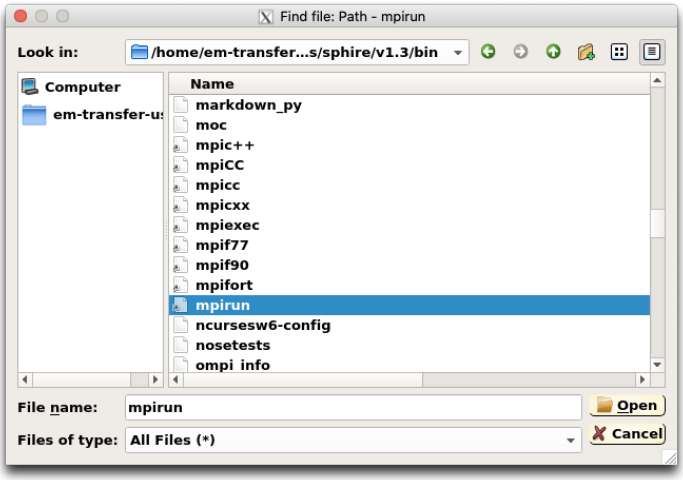


2020-09-03 at 11.01.541.png

Provide the full PATH to the executables. You can press **Ctrl + Return** while editing to open a *File Open Dialog*. If you are not sure where the files are located, but you can execute them in the terminal type:

```
>>> which desired_command
```

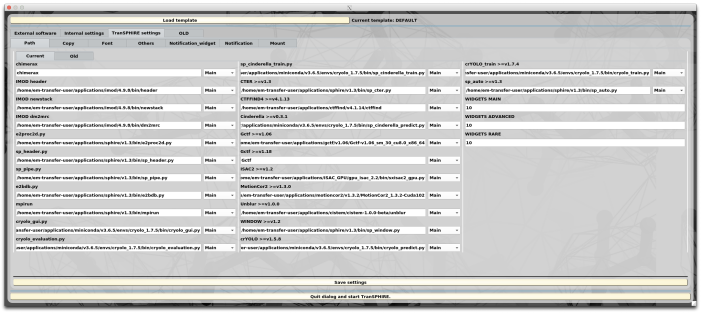
Otherwise contact your system administrator.



2020-09-03 at 11.57.131.png

Note: Please provide the mpirun command that is shipped with the SPHIRE installation.

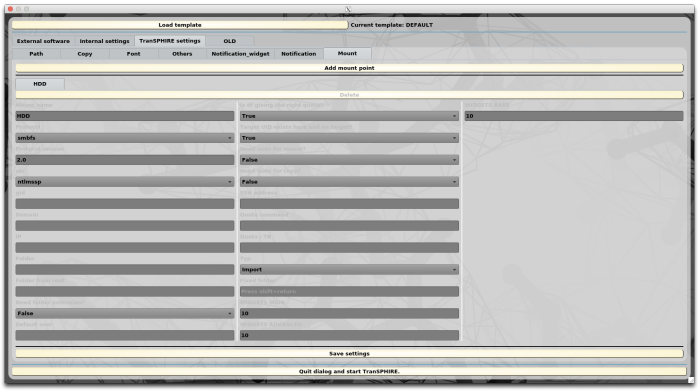
After the information has been provided, click the **Save settings** button.



2020-09-03 at 12.14.361.png

Mount points

Next provide the mount point information to tell TranSPHIRE where possible data is located or where to copy created data to. Click on **TranSPHIRE settings -> Mount**. By default there is a mount point for external hard drives present, but additional mount points need to be added in order to function properly. To mount external Machines, Linux mount.mount_protocol executables are used.



2020-09-03 at 12.47.461.png

There are two possible ways to deal with mount points: *Fixed folder mount points* and *On demand mount points*.

Fixed folder mount points

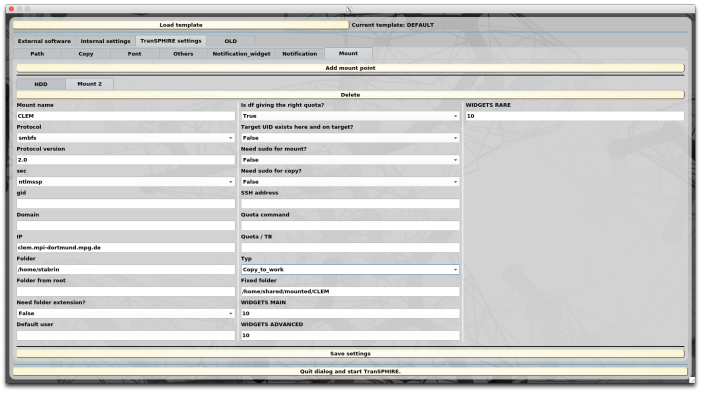
Click on the **Add mount point** button and a new mount point entry `Mount 2` appears. You need to provide the following entries:

- **Mount name**
  - Name of the mount point within the TranSPHIRE GUI.
- **IP**
  - IP adress of the remote device. This is necessary to assure the correct execution of the Auto3D command via SSH.
- **Folder**
  - Mount entry folder name of the remote device.
- **Typ**
  - Choose if the mount point is used to import the data or if it is a destination for either processing or backup.
- **Fixed folder**
  - Folder of the fixed mount point on the local device.

Note: Our cluster is called `CLEM` and is mounted on the local device at the location `/home/shared/mounted/CLEM`. The IP is `clem.mpi-dortmund.mpg.de` and the mounted folder is `/home/stabrin`. Because it is a cluster for data processing, the Typ is `Copy_to_work`. The Fixed folder location is `/home/shared/mounted/CLEM`.

Therefore our configuration is:

- **Mount name**: CLEM
- **IP**: clem.mpi-dortmund.mpg.de
- **Folder**: /home/stabrin
- **Typ**: Copy_to_work
- **Fixed folder**: /home/shared/mounted/CLEM



2020-09-03 at 15.20.491.png

On demand mount points

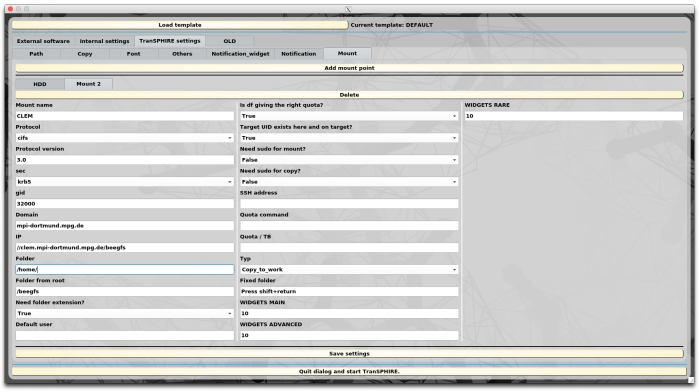
Click on the **Add mount point** button and a new mount point entry `Mount 2` appears. To fill out the respective entries, you should talk to your system administrator. You need to provide the following entries:

- **Mount name**
  - Name of the mount point within the TranSPHIRE GUI.
- **Protocol**
  - Mount protocol.
- **Protocol version**
  - The version of the protocol.
- **sec**
  - The security protocol used for the mount point. If your `sec` value is `krb5`, the cruid option is automatically set to the user. If you need a different behaviour, please contact markus.stabrin@mpi-dortmund.mpg.de.
- **gid**
  - The mount group.
- **Domain**
  - Domain of the mount point.
- **IP**
  - IP adress of the remote device. This is necessary to assure the correct execution of the Auto3D command via SSH.
- **Folder**
  - Mount entry folder name of the remote device.
- **Folder from root**
  - Path to the folder specified in `Folder` from the root directory of the remote device.
- **Need folder extension?**
  - Set to True, to allow dynamic point entries. See the note for an example.
- **Default user**
  - Default user to fill in for mounting. This way only the password needs to be provided. Useful for computers where the mount user does not change.
- **Is df giving the right quota?**
  - The Linux command `df` provides information about the disc occupancy of mount points. However, for file systems that use a quota management this value is usually not correct. If in doubt, leave the settings to `True`.
- **Target UID exists here and on target?**
  - For domain user managed computers. If set to `True`, the provided user/password combination will be used to run an `ls` command on the local machine to provide a sanity check.
- **Need sudo for mount?**
  - **WARNING**: Providing root passwords is not ideal. We would recommended to use `cifs` mount points and allow for password-less sudo rights for `mount.cifs`.

- **Need sudo for copy?**
  - **WARNING**: Use this with caution, a shared account for the TranSPHIRE runs is not recommended. If you use a shared "Transfer" account for the dedicated TranSPHIRE machine, the root password needs to be provided at the beginning of the session to allow for copy of the data to the mount points.
- **SSH address**
  - SSH address used to calculate the quota if `Is df giving the right quota` is set to False.
- **Quota command**
  - Command to calculate the quota on the remote device.
- **Quota / TB**
  - **Deprecated** will be removed in the next versions.
- **Typ**
  - Choose if the mount point is used to import the data or if it is a destination for either processing or backup.
- **Fixed folder**
  - Folder of the fixed mount point on the local device.

Note: The current user is `stabrin` and the authentification mechanism works with a `kerberos ticket`. Our cluster is called `CLEM`. The mount protocol is `cifs` and the version is `3.0`. The sec protocol is `krb5`. `stabrin` is a member of the group `32000` ad the domain is `mpi-dortmund.mpg.de`. The mount IP entry point is `//clem.mpi-dortmund.mpg.de/beegfs` and the mount folder is `/home/`. The path from root to the mount folder is `/beegfs` resulting in `/beegfs/home/` on the remote device. Since every user has its own separate home directory on `CLEM` and we want to allow for dynamic mounting, we have `Need folder extension?` set to `True`. This way we can provide `stabrin` as the folder extension during mounting to mount `/beegfs/home/stabrin`. Additionally, the `Target UID exists here and on target` and we change the setting to `False` as well as the `Typ` to `Copy_to_work`.

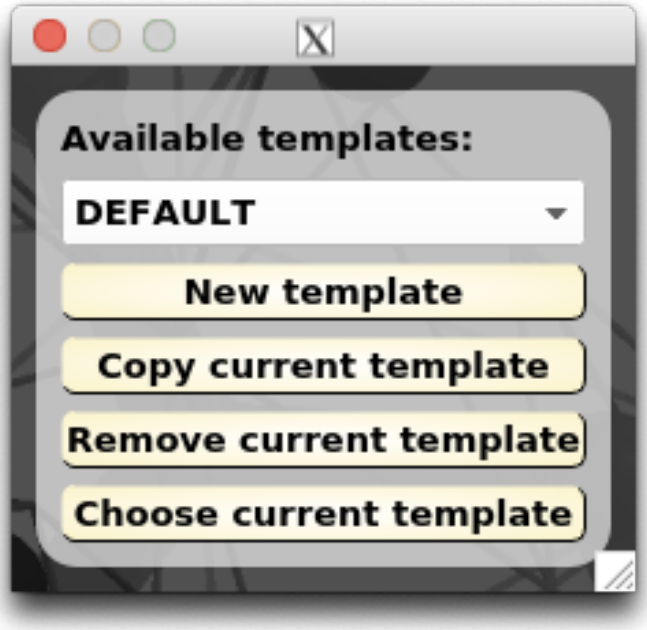
Therefore our configuration is:



2020-09-03 at 16.15.171.png

Create templates

In order to create setting templates you need to click the **Load template** button.



2020-09-03 at 16.39.591.png

- Drop-down widget: You can choose the template here.
- New template: You can create a new and empty template entry.
- Copy current template: Create a new template, but the settings are identical to the template chosen in the drop-down widget.
- Remove current template: Remove the currently selected template.
- Choose current template: Choose the current template in order to change its settings.

Click on **New template** and provide a name like `Tutorial_template` for your new template and set **Choose current template**. The text next to the **Load template** button indicates that the correct template is active.

2020-09-03 at 16.53.521.png 

Now adjust the settings to match the needs for your facility. This helps especially beginner users to avoid making errors during setup.

9.5.3 TranSPHIRE outputs

TranSPHIRE output folder structure

The TranSPHIRE output directory looks like this

Projects
|
|-Project directory 1
|
|-TranSPHIRE_results
|
|-000_Feedback_results
|

|-004_{PICKING_NAME}_feedback_XX
|-005_{EXTRACT_NAME}_feedback_XX
|-006_{2D_CLASSIFICATION_NAME}_feedback_XX
|-007_{2D_CLASS_SELECTION_NAME}_feedback_XX
|-008_{RETRAIN_NAME}_feedback_XX
|-009_{3D_NAME}_feedback_XX
|-000_Import
|-000_Import_meta
|-000_Session_meta
|-001_{COMPRESS_NAME}
|-002_{MOTION_NAME}
|-003_{CTF_NAME}
|-004_{PICKING_NAME}
|-005_{EXTRACT_NAME}
|-006_{2D_CLASSIFICATION_NAME}
|-007_{2D_CLASS_SELECTION_NAME}
|-009_{3D_NAME}
|-XXX_Error_files
|-XXX_Log_files
|-XXX_Filtered_Images
|-XXX_Queue_files
|-XXX_Restart_Backup
|-XXX_Settings
|-XXX_Tar_file_folder
|-Valid_micrograph_info.txt
|-Discarded_micrograph_info.txt
|-{CTF_NAME}_transphire_ctf_partres.txt
|-{CTF_NAME}_transphire_ctf.star
|-{MOTION_NAME}_transphire_motion.txt
|-{MOTION_NAME}_transphire_motion.star
|-{MOTION_NAME}_transphire_motion_relion3.star

...
|-Project directory N

TranSPHIRE output folders

Output folder	Content
Projects	Folder that contains all TranSPHIRE projects. This folder is provided within the TranSPHIRE GUI.

Continued on next page

Table 1 – continued from previous page

Output folder	Content
Project directory X	TranSPHIRE project directory. Every project is created with the provided *Project name*. — Every path of the major output files is stored relative to this directory. **Therefore it is recommended to use this folder as a project directory for further processing so that problems with for example particle polishing can be avoided.**
TranSPHIRE_results	Folder containing the actual TranSPHIRE results.
000_Import	Folder containing the incoming movies. If the incoming movies are already tiff files the files will remain in this folder.
000_Import_meta	Folder containing the meta data that comes with the data. This includes xml, jpg, and spot overview mrc files.
000_Session_meta	Folder containing the meta data that is created by the data aquisition software, but is not the meta data for an acquired image but present. in the specified directory.
001_{COMPRESS_NAME}	Folder containing the compressed movies. — If the incoming movies are already tiff files or no compression is specified, the movies remain in the 000_Import folder. The COMPRESS_NAME depends on the specified compression method.
002_{MOTION_NAME}	Folder containing the results of motion correction. — **In addition to the outputs produced by the chosen program, TranSPHIRE also creates _meta.star files that can be used to run particle polishing in RELION.** The MOTION_NAME depends on the specified motion correction program and version.
003_{CTF_NAME}	Folder containing the results of the CTF estimation. The CTF_NAME depends on the specified ctf estimaion program and version.
004_{PICKING_NAME}	Folder containing the results of the particle picking job. The PICKING_NAME depends on the specified particle picking program and version.
005_{EXTRACT_NAME}	Folder containing the results of particle extraction. The EXTRACT_NAME depends on the specified particle extraction program and version.
006_{2D_CLASSIFICATION_NAME}	Folder containing the results of 2D classification. The 2D_CLASSIFICATION_NAME depends on the specified 2d classification program and version.

Continued on next page

Table 1 – continued from previous page

Output folder	Content
007_{2D_CLASS_SELECTION_NAME}	Folder containing the results of 2D class selection. The 2D_CLASS_SELECTION_NAME depends on the specified 2d class selection program and version.
009_{3D_NAME}	Folder containing the results of 3D ab-initio reconstructon and 3D refinement. The 3D_NAME depends on the specified 3D program and version.
000_Feedback_results	Folder containing the results of the feedback loop. — Every feedback round has its own output folder indicated by a *_feedback_XX* suffix. The *XX* stands for the respective feedback iteration. Results produced outside the feedback loop will be stored in its respective folders outside the 000_Feedback_results folder.
008_{RETRAIN_NAME}_feedback_XX	Folder containing the results of the training of the used picking model. This folder is only present in the 000_Feedback_results folder. The RETRAIN_NAME depends on the specified retrain program and version.
XXX_Error_files	Folder containing the error files of the TranSPHIRE run.
XXX_Log_files	Folder containing the log files of the — TranSPHIRE run. Log information like the current feedback loop iteration, the current picking threshold and last used file numbers are stored.
XXX_Filtered_Images	Folder containing the filtered images from crYOLO before and after the feedback loop.

Continued on next page

Table 1 – continued from previous page

Output folder	Content
XXX_Queue_files	Folder containing the queue status of the — TranSPHIRE run. There are 3 queue files per job: *NAME*, *NAME_done*, *NAME_list*. The *NAME* file contains the information about the to-be-processed files. The *NAME_done* file contains the information of the already processed files. The *NAME_list* file is only filled for jobs that have an additional internal queue like particle picking, particle extraction, and 2d classification. The content indicates that files are ready to be processed but still wait for a certain condition to be met. In case of particle picking, the program waits for 30 seconds before starting the actual picking run to reduce the overhead of program initialisation. For particle extraction, the program waits until all the results of motion correction, particle picking, and ctf estimation arrived. 2D classification and 3D waits until a certain number of particles is accumulated. Because multiple files depend on a different number of input files, the provided queue status can be larger than the number of imported movies for the respective jobs.
XXX_Restart_Backup	Folder containing the obsolete files due to a restart.
XXX_Settings	Folder containing the settings and files for the current session. — Everytime the *Start* button is pressed, the provided external files and the current setup of TranSPHIRE is saved in a new session folder indicated by the current date and time. Internally, the copied files are used instead of the original ones.
XXX_Tar_file_folder	Folder containing the tar files that are created prior to copying when the *Tar to work* or *Tar to backup* option is activated.
Valid_micrograph_info.txt	File containing the extracted meta data for each movie in a star file format. Only the movies that do not violate the range provided by the user are stored in the Valid version.
Discarded_micrograph_info.txt	File containing the extracted meta data for each movie in a star file format. Only the movies that do violate the range provided by the user are stored in the Discarded version.
{CTF_NAME}_transphire_ctf_partres.txt	File containing the CTF estimation information in the SPHIRE partres format. This file can be used to skip CTF estimation in a real processing scenario.
{CTF_NAME}_transphire_ctf.star	File containing the CTF estimation information in the RELION star file format. This file can be used to skip CTF estimation in a real processing scenario.

Continued on next page

Table 1 – continued from previous page

Output folder	Content
{MOTION_NAME}_transphire_motion.txt	File containing a list of valid micrograph entries.
{MOTION_NAME}_transphire_motion.star	File containing micrograph information like the name and path of the DW and non DW summed image.
{MOTION_NAME}_transphire_motion_relion3.star	File containing micrograph information like the name and path of the DW and non DW summed image. Additionally, information to run particle polishing is available. Provide this file to run as input for particle polishing.

9.5.4 TRPC4 tutorial data set

To demonstrate how TranSPHIRE works, a TRPC4 tutorial test data set can be downloaded here: https://ftp.gwdg.de/pub/misc/sphire/TranSPHIRE/TranSPHIRE_1.5_trpc4_tutorial.zip

Unzip the downloaded file `TranSPHIRE_1.5_trpc4_tutorial.zip`:

```
>>> tar -xf TranSPHIRE_nature_communications.zip
```

This extracts a folder called `TranSPHIRE_1.5_trpc4_tutorial.zip`.

The following instructions can also be found in the included `README.rst`.

Operating systems

The software has been tested on the following operating systems:

Ubuntu 18 Ubuntu 20 CentOS 7

Software dependencies

TranSPHIRE is dependent on different programs:

Included in this zip file

|TranSPHIRE| version 1.5.0
SPHIRE version 1.3_transphire
SPHIRE GPU ISAC version 1.1

Not included in this zip file

Anaconda / Miniconda environment – https://www.anaconda.com/
IMOD version 4.9 – https://bio3d.colorado.edu/imod/
CHIMERAX version 1.0 – https://www.cgl.ucsf.edu/chimerax/
MotionCor2 version 1.3.2 – https://emcore.ucsf.edu/ucsf-software
Unblur cisTEM version 1.0.0-beta – https://cistem.org/
CTFFIND4 version 4.1.14 – https://grigoriefflab.umassmed.edu/ctf_estimation_ctffind_ctftilt
GCtf version 1.06 – https://www2.mrc-lmb.cam.ac.uk/research/locally-developed-software/zhang-software/
SPHIRE crYOLO version 1.7.4 – http://cryolo.readthedocs.io/

SPHIRE Cinderella version 0.7.0 – http://sphire.mpg.de/wiki/doku.php?id=auto_2d_class_selection

Required non-standard hardware

- Nvidia GPU

Installation

The installation of TranSPHIRE takes about 5 to 20 minutes, depending on the download speed.

Install SPHIRE_v1.3_transphire

```
>>> bash ./install_sphire_v1.3_transphire.sh
```

Install GPU ISAC

Make sure that you have CUDA available then run:

```
>>> bash ./install_gpu_isac.sh
```

Create a new conda environment for TranSPHIRE

```
>>> bash ./install_transphire.sh
```

Put the TranSPHIRE installation in your PATH

```
>>> export PATH=$(realpath sphire_v1.3_transphire/envs/transphire/bin):${PATH}
```

Install other dependencies

Please install the other dependencies from the *Software dependencies* section.

Demo

A TRPC4 Demo data set containig 120 micrograph movies is coming within the ZIP file.

The expected output can be found in the *TRPC4_demo_results_expected* folder.

The expected runtime of the demo data is:

- 4.5 hours without 3D refinement on 6 cores.
- 5.5 hours with 3D refinement on 6 cores.

On a "normal" GPU machine:

- Intel(R) Xeon(R) CPU E5-2643 v4 @ 3.40Ghz
- 6 cores / hyperthreading 12 cores
- 128 GB RAM
- 2x GeForce RTX 2080 Ti

Instructions

A more detailed version of the instructions is currently in preparation at *transphire.readthedocs.io*.

1. Open the TranSPHIRE GUI

```
>>> transphire --root_directory $(realpath .)
```

2. Click the **Settings** tab.
3. Click the **Input** tab.
   - *Input project path for frames*: click the folder icon and choose the *TRPC4_demo* folder
   - *Input project path for jpg*: click the folder icon and choose the *TRPC4_demo* folder
   - *Input frames extension*: tiff
   - *number of frames*: 50
4. Click the **Output** tab.
   - *Project name*: TRPC4_demo_results
   - *Rename prefix*: **TRPC4_**
   - *Rename suffix*: _demo
5. Click the **Global** tab.
   - *Pixel size*: 0.85
   - *Cs*: 0.001
   - *Gain*: click the folder icon and choose the *TRPC4_demo/gain_ref.mrc* file
   - *Protein radius*: 120
6. Click the **Copy** tab.
   - *Delete data after Import*: Symlink
7. Click the **Path** tab.

   Provide the file path values from your downloaded dependencies.
8. Click the **Motion** tab.
   - *-FmDose*: 1.77
   - *-Patch*: 5 5 20
9. Click the **CTF** tab.
   - *–f_start*: 40
   - *–f_stop*: 4
10. Click the **Picking** tab.
    - *–conf**: click the folder icon and choose the *TRPC4_demo/config_2020_07.json* file
    - *–weights**: click the folder icon and choose the *TRPC4_demo/gmodel_phosnet_202005_N63_c17.h5* file
11. Click the **Class2d** tab and go to the **Advanced** tab.
    - *Nr. Particles*: 5000
    - *–img_per_grp*: 50

- *–minimum_grp_size*: 30
- *MPI processes*: Choose the number of your physical cores available.

12. Click the **Select2d** tab.

- *–weights*: click the folder icon and choose the *TRPC4_demo/config_2020_07.json* file

13. Click the **Auto3d** tab.

    - *–mpi_procs*: Choose the number of your physical cores available.
    - *–mpi_submission_command*: bash
    - *–mpi_submission_template*: click the folder icon and choose the *submission_bash_template.sh* file
    - *–memory_per_node*: Adjust depending on your system
    - *–mol_mass*: 900
    - *–symmetry*: c4
    - *input_volume*: click the folder icon and choose the *TRPC4_reference.hdf* file
    - *Use SSH*: False

    Advanced

    - *Minimum classes*: 0
    - *Minimum particles*: 0

14. Click the **Start** button.

9.6 FAQ

In addition to the TranSPHIRE related questions, we provide an in-depth SPHIRE tutorial on our SPHIRE website: Tutorial. There, not only SPHIRE related topics are discussed, but also general tips and tricks for Single Particle Analysis.

9.6.1 Frames and jpg/meta files are stored in different locations, is that a problem?

No, TranSPHIRE offers the possibility to provide a path to the frames and a path to the jpg/meta data. If you setup the TranSPHIRE session after the first images has been collected, TranSPHIRE will try to search for the respective directories itself.

9.6.2 How can I handle multiple gain references in one session?

TranSPHIRE assigns the provided gain reference while finding new files and assigns the currently used gain with the found image. Therefore, every input image is linked to it's respective gain reference. To set a new gain reference for new images, press the **Stop** button, provide the new gain reference, and click the **Start** button to continue.

9.6.3 What is the output folder structure of TranSPHIRE?

For a detailed explanation about the output folders of TranSPHIRE, please visit *TranSPHIRE outputs*.

9.6.4 How to create a template for faster setup?

You can find more information here: *Basic TranSPHIRE setup*.

9.6.5 What kind of computer should I use?

TranSPHIRE is designed to run on a Linux system. The better the hardware is, the faster the processing will be. This also allows to stay on-the-fly for faster data acquisition schemes.

You can find more information here: *Hardware recommendations*.

9.6.6 How do I measure my particle radius in pixels?

There are mutliple ways of doing this, but we recommend to use *e2display.py* from the SPHIRE package.

```
>>> e2display.py example_image.mrc
```

Press the **Middle mouse button** and the **Meas** tab. Keep the `A/Pix` value to `1.0` and click+drag a line on the micrograph. The `Len` value shows the distance in pixels.

If the particles do not have a globular shape you should choose a radius that is a wider than the measured one to allow for a more liberal centering.

9.6.7 What box size should I use?

A list of good box sizes can be found on the EMAN2 box size recommendation website.

The recommendation is to use a box size of 1.5x to 2x of the longest particle axis (3x to 4x of the radius).

By default, TranSPHIRE will use the next bigger value of a "good" box size from the list after multiplying the provided protein radius by 3.

Note: Example 1:

particle_radius = 100
box_size = 300 # particle_radius * 3.
final_box_size = 300 # 300 is in the list of good values.

Example 2:

particle_radius = 101
box_size = 303 # parrticle_radius * 3.
final_box_size = 320 # 303 is not in the list of good values, the next larger good value is 320.

9.6.8 How can I see how many particles have been extracted?

To check the number of extracted particles click on

Visualisation -> Plot Extract -> Plot per micrograph -> accepted (lower tab row)

Check the **Sum:** field of the data statistics area.

9.6.9 How do I cite TranSPHIRE?

To cite TranSPHIRE use the following citation: Stabrin, M., Schoenfeld, F., Wagner, T. et al. TranSPHIRE: automated and feedback-optimized on-the-fly processing for cryo-EM. Nat Commun 11, 5716 (2020). https://doi.org/10.1038/s41467-020-19513-2

Please also properly cite the individual tools that you used during the TranSPHIRE run.

9.7 How to contribute

There are several ways how one can contribute to the TranSPHIRE project.

- Bug reports
- Feature requests
- Missing documentation
- Nice words

If you are interested to contribute, feel free to write an e-mail to markus.stabrin@mpi-dortmund.mpg.de.

9.8 transphire package (Developer area)

9.8.1 Subpackages

transphire.support_scripts package

Submodules

transphire.support_scripts.chimerax module

Module contents

9.8.2 Submodules

9.8.3 transphire.buttoncontainer module

9.8.4 transphire.emaildialog module

TranSPHIRE is supposed to help with the cryo-EM data collection Copyright (C) 2017 Markus Stabrin

This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program. If not, see <http://www.gnu.org/licenses/>.

class transphire.emaildialog.**EmailDialog**(*parent=None*)

Bases: PyQt5.QtWidgets.QDialog

EmailDialog widget.

Inherits from: QDialog

Buttons: OK Cancel

LineEdit: Name - Name, that associates with the E-Mail E-Mail - E-Mail adress of the user

get_email()

Return the text of the E-Mail.

Arguments: None

Return: E-Mail

get_name()

Return the text of the Name.

Arguments: None

Return: Name

9.8.5 transphire.framecontainer module

TranSPHIRE is supposed to help with the cryo-EM data collection Copyright (C) 2017 Markus Stabrin

This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program. If not, see <http://www.gnu.org/licenses/>.

class transphire.framecontainer.**FrameContainer**(*parent=None*, ***kwargs*)

Bases: PyQt5.QtWidgets.QWidget

FrameContainer widget.

Inherits from: QWidget

Buttons: Add - Add FrameWidget widget to the layout

LineEdit: First frame - First frame to use for a smaller amount of dose Last frame - Last frame to use for a smaller amount of dose

add_widget()
Add a FrameWidget widget to the dynamic layout. Called when the add button is clicked.

Arguments: None

Returns: None

enable(*var*, *use_all*)
Enable or disable the buttons and widgets.

Arguments: var - State of buttons (True or False) use_all - Disable all buttons (True) or only some (False)

Return: None

get_settings()
Get settings of all FrameWidget widgets.

Arguments: None

Returns: settings - List of FrameWidget settings

set_settings(*settings*)
Add FrameWidget widgets to the dynamic layout based on setting entries.

Arguments: settings - List of FrameWidget settings.

Returns: None

9.8.6 transphire.framewidget module

TranSPHIRE is supposed to help with the cryo-EM data collection Copyright (C) 2017 Markus Stabrin

This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program. If not, see <http://www.gnu.org/licenses/>.

class transphire.framewidget.**FrameWidget**(*first*, *last*, *dose_weight*, *default*, *parent=None*)
Bases: PyQt5.QtWidgets.QWidget

FrameWidget widget.

Inherits from: QWidget

Buttons: Delete - Delete this instance of the variable.

Signals: delete - Delete this instance of the variable (class instance).

delete

get_settings()
Get the settings

Arguments: None

Return: settings as dictionary

9.8.7 transphire.inputbox module

TranSPHIRE is supposed to help with the cryo-EM data collection Copyright (C) 2017 Markus Stabrin

This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program. If not, see <http://www.gnu.org/licenses/>.

class transphire.inputbox.**InputBox**(*is_password*, *restart_names=None*, *is_stop=False*, *settings=None*, *parent=None*)

Bases: PyQt5.QtWidgets.QDialog

Show a message box with an input field.

Inherits: QDialog

Signals: None

getText()

Get the text from the label.

Arguments: None

Returns: Text content

get_restart_dict()

handle_check(*state*)

setDefault(*text*)

Set the default values for the QLineEdit.

text - Text to put

Returns: None

setText(*heading*, *text*)

Set the text to the label.

Arguments: heading - Heading of the window text - Text of the label

Returns: None

set_type(*this_type*)

9.8.8 transphire.loadcontent module

TranSPHIRE is supposed to help with the cryo-EM data collection Copyright (C) 2017 Markus Stabrin

This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program. If not, see <http://www.gnu.org/licenses/>.

class transphire.loadcontent.**LoadContent**(*typ*, *separator*, *settings_folder*, *hdd=None*, *parent=None*)

Bases: PyQt5.QtWidgets.QWidget

LoadContent widget. Widget used for the LoadContentContainer.

Inherits from: QWidget

Signals: delete - Emited, when the delete button is pressed (object)

delete

get_settings()

Get the settings from the child widgets.

Arguments: None

Return: List of settings

set_settings(*settings*)

Set settings to the entry widgets.

Arguments: settings - List of settings

Return: None

9.8.9 transphire.loadcontentcontainer module

TranSPHIRE is supposed to help with the cryo-EM data collection Copyright (C) 2017 Markus Stabrin

This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program. If not, see <http://www.gnu.org/licenses/>.

class transphire.loadcontentcontainer.**LoadContentContainer**(*typ*, *template_name*, *templates*, *settings_folder*, *is_shared*, *default_file*, *parent=None*)

Bases: PyQt5.QtWidgets.QWidget

LoadContentContainer widget Inherits from: QWidget

Buttons: Save settings - Save currently defined settings to settings files Add - Add new Mount entry (Case Mount)

add_widget(*name=None*, *hdd=None*)

Add new widget to layout.

Arguments: name - Name of new mount point (default None) hdd - Is hdd entry (default None)

Return: None

copy_from_template(*template*)

get_settings()
Get settings from tab or normal normal widget.

Arguments: None

Return: List of widget settings

get_settings_tab()
Get settings from tab widget.

Arguments: None

Return: List of widget settings

get_settings_widget()
Get settings from normal widget.

Arguments: None

Return: List of widget settings

remove_widget(*separator*)
Remove widget from layout.

Arguments: separator - List of seperator widgets

Return: None

save_settings()
Save settings specified in widget to json text file.

Arguments: None

Return: None

set_settings(*settings*)
Set settings to widget.

Arguments: settings - List of widget settings

Return: None

9.8.10 transphire.loadwindow module

TranSPHIRE is supposed to help with the cryo-EM data collection Copyright (C) 2017 Markus Stabrin

This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program. If not, see <http://www.gnu.org/licenses/>.

class transphire.loadwindow.**DefaultSettings**(*apply*, *settings_directory*, *template_name*, *parent=None*)

Bases: PyQt5.QtWidgets.QDialog

DefaultSettings dialog. Dialog used to enter default values.

Inherits from: QDialog

accept_apply()
Set the apply settings variable to True before saving.

Argumens: None

Return: None

add_tab(*widget*, *name*)
Add a new widget to the tab widget

Arguments: widget - Widget to add as new tab Name - Name of the new tab

Return: None

add_tabs()

check_modified_widgets(*done*)
Check, if a widget is modified before saving.

Arguments: done - If True, close after input, else apply settings.

Return: True if no modification, else False

clear_tabs()

closeEvent(*event*)
Handle the close event.

Arguments: event - Close event

Return: None

create_initial_tabs(*tab_dict*, *parent_widget*, *is_old=False*)

fill_default_dict()
Fill the default dict with the information from the function dict.

Arguments: None - The self.default_dict is used

Returns: None - The self.default_dict is changed in-place

get_apply()
Getter for the self.apply variable.

Arguments: None

Return: Content of self.apply

static get_content_default(*edit_settings*, *apply*, *settings_folder*, *template_name*)
Staticmethod to open the default content dialog.

Arguments: edit_settings - If True, open default settings dialog, else just return content apply - Apply settings after closing the default settings dialog settings_folder - Folder to store the default settings

Return: Content for the widgets, Content of the apply variable

is_in_content(*tab_dict*, *name*)

load_template()

recursive_clear(*tab_widget*)

9.8.11 transphire.logviewer module

class transphire.logviewer.**LogViewer**(*show_indicators=False*, *indicator=''*, *file_name=''*, *parent=None*)

Bases: PyQt5.QtWidgets.QWidget

appendPlainText(*text*, *indicator='log'*, *user=False*)

change_state(*state*)

get_indicator(*indicator*)

increment_indicator(*indicator*, *text=''*)

my_click_event(*event=None*)

reset_plain_text(*text*)

set_project_path(*project_path*, *log_path*, *error_path*)

submit_text()

update_plain_text(*force=False*)

9.8.12 transphire.logviewerdialog module

class transphire.logviewerdialog.**LogViewerDialog**(*parent=None*)

Bases: PyQt5.QtWidgets.QDialog

add_tab(*widget*, *name*)

9.8.13 transphire.mainwindow module

TranSPHIRE is supposed to help with the cryo-EM data collection Copyright (C) 2017 Markus Stabrin

This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program. If not, see <http://www.gnu.org/licenses/>.

class transphire.mainwindow.**MainWindow**(*content_raw*, *content_gui*, *content_pipeline*, *settings_folder*, *mount_directory*, *template_name*, *version*, *n_feedbacks*, *parent=None*)

Bases: PyQt5.QtWidgets.QMainWindow

MainWindow widget. Connects GUI and worker threads.

Inherits from: QMainWindow

Buttons: None

Signals: None

abort_finished(**args*, ***kwargs*)

Set the mount worker abort variable to True.

Arguments: None

Return: None

check_quota()
Check the quota for the project and scratch directory.

Arguments: None

Return: None

closeEvent(*event*)
Quit threads before close and check if the process is still running

Arguments: event - QCloseEvent.

Return: None

continue_dialog(*text1*, *text2*, *is_stop=False*, *is_restart=False*, *settings=None*)
Check if the user wants to run the continue mode.

Arguments: text1 - Dialog window name. text2 - Text of the dialog.

Return: True, if the input is YES!

enable(*var*, *use_all=False*)
Enable or disable widgets

Arguments: var - Enable status of the widgets. use_all - Disable/Enable everything (Default False)

Return: None

fill_content(*content_gui*)
Fill the layouts of the central widget.

Arguments: content_gui - Content used to create the GUI outfit.

Return: List of errors that occured.

get_start_settings(*monitor=False*)
Start TranSPHIRE processing.

Arguments: None

Return: None

hide_tab(*sender*, *text*)

load(*file_name=None*)
Load settings from settings file.

Arguments: file_name - Name of the file (default None)

Return: None

monitor(*start*)
Start the TranSPHIRE monitor processing.

Arguments: start - True if start, False if stop

Returns: None

new_round_plot()

postprocess_content(*error_list*)
Do postprocessing of creating GUI content, like connecting signals.

Arguments: error_list - List of errors that occured.

Return: True, if saving was succesful.

reset_gui(*template_name*, *load_file*)
Reset the content of the mainwindow.

Arguments: template_name - Name of the template to load load_file - Settings file (default None).

Return: None

save(*file_name=None*, *temp=False*, *interactive=False*, *do_message=True*)
Save GUI status to file.

Arguments: file_name - File name to save settings to. temp - File is a temporary save file.

Return: True, if saving was succesful.

save_temp_settings()
Save the status of the GUI in a temp file.

Arguments: None

Return: True, if saving was succesful.

set_central_widget()
Reset the central widget of the MainWindow.

Arguments: None

Return: None

set_design(*settings*)
Load settings from settings file.

Arguments: settings - Settings as dictionary.

Return: None

set_layout_structure()
Setup the layout structure for the central widget.

Arguments: None

Return: None

set_settings(*settings*)
Load settings from settings file.

Arguments: settings - Settings as dictionary.

Return: None

set_visualisation()

static settings_to_dict(*settings*)
Make the settings readable for the widgets set settings method.

Arguments: settings - Settings as dictionary.

Return: None

sig_reset

start()

start_threads(*content_pipeline*)
Start threads used in TranSPHIRE.

Arguments: content_pipeline - Content used to start processing threads.

Return: None

stop(*abort=False*)
Stop the process.

Arguments: None

Return: None

stop_dialog()
Check if the user really wants to stop the process.

Arguments: None

Return: None

9.8.14 transphire.messagebox module

TranSPHIRE is supposed to help with the cryo-EM data collection Copyright (C) 2017 Markus Stabrin

This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program. If not, see <http://www.gnu.org/licenses/>.

class transphire.messagebox.**MessageBox**(*is_question*, *parent=None*)
Bases: PyQt5.QtWidgets.QDialog

Show a message box

Inherits: QDialog

Signals: None

setDefault(*text*)

setText(*heading*, *text*)
Set the text to the label.

Arguments: heading - Heading of the window text - Text of the label

Returns: None

9.8.15 transphire.mountcalculator module

TranSPHIRE is supposed to help with the cryo-EM data collection Copyright (C) 2017 Markus Stabrin

This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program. If not, see <http://www.gnu.org/licenses/>.

class transphire.mountcalculator.**MountCalculator**(*name*, *parent=None*)
Bases: PyQt5.QtCore.QObject

MountCalculator object.

Inherits from: QObject

Buttons: None

Signals: sig_finished - Signal emitted, if process finished (str, str, str)

calculate_df_quota(*key*, *mount_folder*)
Calculate the quota with the help of the df command.

Arguments: key - Mount point key mount_folder - Mount folder

Return: None

calculate_get_quota(*key*, *quota*, *mount_folder*)
Calculate the quota by calculating the size of every file.

Arguments: key - Mount point key mount_folder - Mount folder quota - User provided maximum quota

Return: None

calculate_ssh_quota(*user*, *folder*, *device*, *mount_folder*, *ssh_dict*, *quota_command_dict*, *password_dict*)
Calculate the quota via ssh.

Arguments: mount_folder - Mount folder user - Username folder - Folder to mount device - Device name ssh_dict - ssh_dict containing ssh information quota_command_dict - Dictionary containing the quota commands password_dict - Dictionary containing passwords

Return: None

get_folder_size(*folder*, *size*)
Get the size of the folder recursively

Arguments: folder - Folder to check contents size - Current caclulated size

Return: Calculated size

static get_quota_quota_command(*text*, *folder*)
Extract the quota from the quota command.

Arguments: text - Text returned by quota command. folder - Mounted folder

Return: None

get_ssh_quota(*user*, *folder*, *device*)
Get the quota via ssh command.

Arguments: user - User name folder - Mounted folder device - Device name

Return: None

sig_finished

9.8.16 transphire.mountcontainer module

TranSPHIRE is supposed to help with the cryo-EM data collection Copyright (C) 2017 Markus Stabrin

This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program. If not, see <http://www.gnu.org/licenses/>.

class transphire.mountcontainer.**MountContainer**(*content_mount*, *mount_worker*, *parent=None*, ***kwargs*)

Bases: PyQt5.QtWidgets.QWidget

MountContainer widget.

Inherits from: QWidget

Buttons: None

Signals: None

enable(*var*, *use_all*)

Enable or disable the content.

Arguments: var - State of buttons (True or False) use_all - Disable all buttons (True) or only some (False)

Return: None

get_settings()

Return settings of the container.

Arguments: None

Return: Settings as list

set_threadlist(*thread_list*)

Set the thread instances for the different objects.

Arguments: thread_list - List of mount threads

Return: None

9.8.17 transphire.mountwidget module

TranSPHIRE is supposed to help with the cryo-EM data collection Copyright (C) 2017 Markus Stabrin

This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program. If not, see <http://www.gnu.org/licenses/>.

class transphire.mountwidget.**MountWidget**(*content*, *mount_worker*, *parent=None*)

Bases: PyQt5.QtWidgets.QWidget

MountWidget widget.

Inherits from: QWidget

Buttons: mount_button - Mount mount point umount_button - Umount mount point

Signals: None

get_settings()
Return settings of the content.

Arguments: None

Returns: Settings as dictionary

mount()
Mount device preparation

Arguments: None

Return: None

set_current_folder(*device*, *folder*)

set_thread_object(*thread_object*)
Set the thread object.

thread_object - Thread object to set.

Returns: None

umount()
Umount device signal preparation

Arguments: None

Return: None

9.8.18 transphire.mountworker module

TranSPHIRE is supposed to help with the cryo-EM data collection Copyright (C) 2017 Markus Stabrin

This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program. If not, see <http://www.gnu.org/licenses/>.

class transphire.mountworker.**MountWorker**(*password*, *settings_folder*, *mount_directory*, *parent=None*)

Bases: PyQt5.QtCore.QObject

Mounting and unmounting shared devices.

Inherits from: QObject

Buttons: None

Signals: sig_mount_hdd - Signal connected to mount HDD (device|str) sig_mount - Signal connected to mount a mount point (device|str, user|str, password|str, folder|str, server|str, typ|str, domain|str, version|str, sec|str, gid|str) sig_umount - Signal connected to unmount a mount point (device_folder|str, device|str, thread|object)

sig_success - Signal emitted, if a task was a success (text|str, device|str, color|str) sig_error - Signal emitted, if a task was a failure (text|str, device|str) sig_info - Signal emitted, to show text in a text box (text|str) sig_notification - Signal emitted, to send a notification message (text|str)

sig_add_save - Signal connected to add a save file to the dictionary (device|str, ss_address|str, quota_command|str, is_right_quota|str, quota|str) sig_load_save - Signal connected to load data from save file (No object) sig_refresh - Signal connected to recalculate quota (No object) sig_quota - Signal emitted, to refresh the quota status in the GUI (text|str, device|str, color|str) sig_set_settings - Signal connected set quota related settings (settings|object)

sig_calculate_ssh_quota - Signal emitted to calculate the quota via ssh (user|str, folder|str, device|str, mount_folder|str, ssh_dict|object, quota_command_dict|object, password_dict|object) sig_calculate_df_quota - Signal emitted to calculate the quota via system information (device|str, mount_folder|str) sig_calculate_get_quota - Signal emitted to calculate the quota brute force (device|str, total_quota|str, mount_folder|str)

add_save(*device*, *ssh_address*, *quota_command*, *is_right_quota*, *quota*)
Add a save file to the dictonaries.

Arguments: device - Mounted device name ssh_address - ssh adress quota_command - Command to calculate quota via ssh is_right_quota - True, if df is showing the right quota quota - Provided maximum quota

Return: None

check_connection()
Check if a mount connection crashed

Arguments: None

Return: None

fill_quota_project_and_scratch(*name*, *directory*, *warning*, *quota_limit*)
Refresh quota information for the project and scratch directory.

Arguments: name - Name (project or scratch) directory - Directory to check warning - current warning status quota_limit - Limit of the quota to show a warning

Return: Current warning status

load_save()
Load connection status from the files

Arguments: None

Return: None

mount(*device*, *user*, *password*, *folder*, *server*, *typ*, *domain*, *version*, *sec*, *gid*, *folder_from_root*, *fixed_folder*)
Mount device except HDD

Arguments: device - Device name user - Username password - User password folder - Mount folder server - Server name typ - Mount type domain - Domain name version - Mount type version sec - security protocol gid - groupid to mount folder_from_root - Absolute path pointing towards the mount point

Return: None

mount_hdd(*device*)
Mount external HDD

Arguments: device - Device name

Return: None

refresh_quota()
Refresh quota information.

Arguments: None

Return: None

set_settings(*settings*)
Set settings used by the worker.

Arguments: settings - TranSPHIRE settings

Return: None

sig_add_save

sig_calculate_df_quota

sig_calculate_get_quota

sig_calculate_ssh_quota

sig_error

sig_info

sig_load_save

sig_mount

sig_mount_hdd

sig_notification

sig_quota

sig_refresh

sig_set_folder

sig_set_settings

sig_success

sig_umount

umount(*device_folder*, *device*, *fixed_folder*, *thread_object*)
Unmount device

Arguments: device_folder - Mount point folder device - Device name thread_object - Thread object that is connected to the mount point

Return: None

transphire.mountworker.**check_existence**(*mount_directory*, *mount_folder*)
Check existence of the mount folder and create it if it does not

Arguments: mount_folder - tolder to check

Return: True, if the mount folder exists

9.8.19 transphire.notificationcontainer module

TranSPHIRE is supposed to help with the cryo-EM data collection Copyright (C) 2017 Markus Stabrin

This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program. If not, see <http://www.gnu.org/licenses/>.

class transphire.notificationcontainer.**NotificationContainer**(*content*, *mount_worker*, *process_worker*, *settings_folder*, *parent=None*, ***kwargs*)

Bases: PyQt5.QtWidgets.QWidget

Container for notification widgets

Inherits: QWidget

Signals: sig_stop - Signal is emitted, if the user is sending /stop via telegram

add_email()
Add a new e-mail adress

Arguments: None

Returns: None

enable(*var*, *use_all*)
Enable or disable the widgets of the widget.

Arguments: var - True(Enable) or False(Disable) use_all - Disable or enable all

Returns: None

get_settings()
Get settings as dict

Arguments: None

Returns: Settings dictionary as list

get_telegram_messages()
Check the messages in telegram for commands.

Arguments: None

Returns: None

get_telegram_user(*page*)
Get the users and the user id

Arguments: page - Webpage of the telegram bot

Returns: Dictionary of users

send_notification(*text*)
Send a notification to specified users.

Arguments: text - Text to send

Returns: None

send_to_user(*user_id*, *text*, *name*)
Send a message to all users.

Arguments: user_id - User id of the receiving person text - Text of the message name - Name of the sender

Returns: None

set_settings(*settings*)
Set settings.

Arguments: settings - Settings as dictionary to set.

Returns: None

sig_stop

update()
Update the combo boxes.

Arguments: None

Returns: None

update_email()
Update E-Mail file.

Arguments: None

Returns: None

update_telegram()
Update telegram settings file

Arguments: None

Returns: None

9.8.20 transphire.notificationwidget module

TranSPHIRE is supposed to help with the cryo-EM data collection Copyright (C) 2017 Markus Stabrin

This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program. If not, see <http://www.gnu.org/licenses/>.

class transphire.notificationwidget.**NotificationWidget**(*name*, *default*, *default_programs_dict*, *parent=None*)

Bases: PyQt5.QtWidgets.QWidget

Widget for notification phone numbers

add_exceptions(*name*)
Add a person to the exception list and dont send notification anymore.

Arguments: name - Name of the person

Returns: None

change_tooltip(*text*)

clear_combo()
Remove all users from the combo box.

Arguments: None

Returns: None

get_settings()
Get text of the currently selected combo item.

Arguments: None

Returns: Settings dictionary

set_settings(*name*, *state*)
Set currently selected combo item in text.

Arguments: name - Name of the person that should be currently selected. state - State of the person (True/False;Enables/Disables)

Returns: None

update_combo(*typ*, *users*)
Update the combo boxes.

Arguments: users - User dictionary

Returns: None

9.8.21 transphire.passworddialog module

TranSPHIRE is supposed to help with the cryo-EM data collection Copyright (C) 2017 Markus Stabrin

This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program. If not, see <http://www.gnu.org/licenses/>.

class transphire.passworddialog.**PasswordDialog**(*folder*, *default*, *login*, *extension*, *fixed_folder*, *parent=None*)

Bases: PyQt5.QtWidgets.QDialog

User password dialog

Inherits: QDialog

Signals: None

9.8.22 transphire.plotcontainer module

TranSPHIRE is supposed to help with the cryo-EM data collection Copyright (C) 2017 Markus Stabrin

This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program. If not, see <http://www.gnu.org/licenses/>.

class transphire.plotcontainer.**PlotContainer**(*name*, *content*, *plot_labels*, *plot_name*, *plot_worker*, *plot_type*, *layout*, **args*, *parent=None*, ***kwargs*)

Bases: PyQt5.QtWidgets.QMainWindow

Qwidget in combination with a FigureCanvas.

Inherits: QMainWindow

activate_tab(*name*)

Activate the tab with the name: name.

Arguments: name - Name of the activation

Returns: None

enable(*var*, *use_all*)

Enable or disable the widgets.

var - It True, enable widgets, else disable use_all - If True, enable/disable all widgets, else only a subset

Returns: None

eventFilter(*source*, *event*)

Override the QMainWindow eventFilter function.

source - Source that led to the event trigger event - Emitted event

Returns: True, if it has been a close event -> Redock widget Event, if it is another event

reset_plot()

select_tab(*widget*)

set_floating()

set_visibility(*visible*, *name*)

synchronize_tabs(*widget*)

update_figure(*name*, *name_no_feedback*, *data*, *directory_name*, *settings*)

Update the figure in the canvas

name - Name of the program that called this function. data - Data to plot. directory_name - Directory to save plots to. settings - TranSPHIRE settings

Returns: None

class transphire.plotcontainer.**TwinContainer**(*dock_widget*, **args*, *parent=None*, ***kwargs*)

Bases: PyQt5.QtWidgets.QWidget

add_to_layout(*name*, *widget*)

handle_show(*name*, *widget*, *state*)

mouse_twin_event(*event*)

9.8.23 transphire.plotwidget module

TranSPHIRE is supposed to help with the cryo-EM data collection Copyright (C) 2017 Markus Stabrin

This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program. If not, see <http://www.gnu.org/licenses/>.

class transphire.plotwidget.**MplCanvas**(*no_grid*, *width=5*, *height=5*, *dpi=100*, *parent=None*)

Bases: matplotlib.backends.backend_qtagg.FigureCanvasQTAgg

sig_twin

class transphire.plotwidget.**MplCanvasWidget**(*no_grid*, *plot_type*, *is_twin=False*, *width=5*, *height=5*, *dpi=100*, *parent=None*)

Bases: PyQt5.QtWidgets.QWidget

update_labels(*title*, *label_x*, *label_y*)

class transphire.plotwidget.**PlotWidget**(*label*, *plot_typ*, *dock_widget*, *twin_container*, **args*, *parent=None*, ***kwargs*)

Bases: PyQt5.QtWidgets.QWidget

PlotWidget widget. Widget used to show data plots.

Inherits from: QWidget

Signals: None

add_canvas()

clear_canvas()

do_data_reset()

force_update(*do_message=False*)

hide_marker()

hide_twin(*state*)

static high_res(*x_data*, *y_data*, *splits*)

prepare_axes(*update*)

set_current_image_name(*text*)

set_settings(*name*, *name_no_feedback*, *data*, *directory_name*, *settings*)

setup_values()

start_plotting()

update_data(*do_message=False*)

update_figure(*do_message=False*)

update_helpers(*canvas*, *plot_idx*, *update*, *plot_type*)

update_histogram(*canvas*, *plot_idx*, *update*)

update_image()

update_image_plot(*canvas*, *data_x*, *data_y*, *high_res*, *label*, *marker*, *color*, *idx*)

update_trim()

update_values(*canvas*, *plot_idx*, *update*)

class transphire.plotwidget.**SelectWidget**(*parent=None*)
Bases: PyQt5.QtWidgets.QWidget

check_enable(*idx*)

filter_combo(*text*)

get_value()

handle_change()

reset_values()

set_values(*value_list*)

sig_update

class transphire.plotwidget.**TrimWidget**(*plot_typ*, *min_default_x*, *max_default_x*, *min_default_y*, *max_default_y*, *bin_default*, *parent=None*)
Bases: PyQt5.QtWidgets.QWidget

get_values()

reset_values()

set_state(*state*)

set_values(*value_dict*)

sig_hide

sig_set_state

sig_update

class transphire.plotwidget.**ViewWidget**(*parent=None*)
Bases: PyQt5.QtWidgets.QWidget

sig_hide

update_label(*label_dict*)

9.8.24 transphire.plotworker module

TranSPHIRE is supposed to help with the cryo-EM data collection Copyright (C) 2017 Markus Stabrin

This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program. If not, see <http://www.gnu.org/licenses/>.

class transphire.plotworker.**PlotWorker**(*parent=None*)
Bases: PyQt5.QtCore.QObject

Plot different information about motion correction and ctf estimation.

Inherits: QObject

Signals: sig_data - Emitted, if data for plotting is found (name|str, data|object, directory|str, settings|object) sig_notification - Emitted, if phase plate limit is reached. (text|str)

calculate_array()
Calculate array.

Returns: None

static calculate_array_now(*name*, *name_no_feedback*, *directory_name*, *settings*)

reset_list()

send_data(*data*)

set_settings(*settings*)
Set settings for the calculation of the arrays.

name - Name of the software that calls the calculation directory_name - Name of the directory that contains the log files settings - TranSPHIRE settings

Returns: None

sig_calculate

sig_data

sig_new_round

sig_reset

sig_reset_list

sig_set_visual

sig_visible

9.8.25 transphire.processthread module

TranSPHIRE is supposed to help with the cryo-EM data collection Copyright (C) 2017 Markus Stabrin

This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program. If not, see <http://www.gnu.org/licenses/>.

class transphire.processthread.**ProcessThread**(*shared_dict*, *name*, *content_settings*, *queue_com*, *password*, *settings*, *mount_directory*, *use_threads_set*, *stop*, *abort*, *has_finished*, *data_frame*, *parent=None*)

Bases: object

Worker thread

Inherits from: object

Buttons: None

Signals: None

add_to_queue(*aim*, *root_name*, *allow_dublicate=False*)
Add item to queue.

Arguments: aim - Aim queue root_name - Name to add

Return: None

add_to_queue_file(*root_name*, *file_name*, *allow_dublicate=False*)
Add item to queue_file.

Arguments: root_name - Name to add file_name - File to append to

Return: Name removed from the queue.

all_in_queue_file(*aim*, *root_name*, *lock=True*)
Add item to queue.

Arguments: aim - Aim queue root_name - Name to add

Return: None

already_in_queue_file(*aim*, *root_name*)
Add item to queue.

Arguments: aim - Aim queue root_name - Name to add

Return: None

already_in_translation_file(*root_name*)
Check, if the root_name already exists in the translation file.

root_name - root_name

Returns: True, if root_name in translation file.

append_to_translate(*root_name*, *new_name*, *xml_file*, *output_queue_dict=None*)
Write to the translation file.

root_name - Root name of the file new_name - New name of the file xml_file - XML or GTG file that contains meta data information

Returns: None

check_connection()
Check, if the process got a lost connection error.

Arguments: None

Return: True, if the process did not get an error, else False

check_full()
Check, if the program got an no space error.

Arguments: None

Return: True, if there is a no space error, else False

static check_if_mounted(*directory*)
Check, if the mount point is still present.

Arguments: directory - Directory to check.

Return: True, if the mount point is still present, else False

check_queue_files(*root_name*)

check_quota()
Check, if the computer reached the quota limit.

Arguments: None

Return: True, if the computer did not hit the limit, else False

check_ready_for_copy(*file_out*)
Check, if the current file is ready to copy.

file_out - File to copy

Returns: True, if ready

copy_as_another_user(*file_in*, *file_out*)
Copy to device as another user via sudo.

file_in - Input file path file_out - Output file path

Returns: None

copy_as_user(*file_in*, *file_out*)
Copy to another device.

file_in - Input file path file_out - Output file path

Returns: None

copy_extern(*my_typ*, *copy_file*)

create_combines(*combine_list*, *output_queue_dict=None*)

static delete_file_to_delete(*file_to_delete*)

file_to_distribute(*file_name*, *output_queue_dict=None*)

static get_gtg_info(*xml_file*, *entries*, *first_entry*)

static get_hash(*file_in*, *chunksize=1048576*)

static get_xml_info(*xml_file*, *entries*, *first_entry*)

lost_connection(*typ*)
Handle lost connection errors.

Arguments: typ - Typ of the error

Return: None

mkdir_p_as_another_user(*folder*)
Create folders recursively as another user with the help of sudo.

folder - Folder structure to create

Returns: None

recursive_search(*directory*, *file_list*, *find_meta*)
Find files in a recursive search.

directory - Directory to search files. file_list - List of files that have been found. find_meta - Find meta data flag.

Returns: List of files found

remove_from_queue()
Remove item from queue.

Arguments: None

Return: Name removed from the queue.

remove_from_queue_file(*root_name*, *file_name*, *lock=True*)
Remove the files from the queue file.

Arguments: root_name - Name of the file to delete

Return: None

remove_from_translate(*root_name*, *output_queue_dict=None*)
Remove line from the translation file.

root_name - Root name of the file to remove

Returns: None

reset_queue(*aim=None*, *switch_feedback=False*, *remove_pattern='THIS IS A DUMMY PATTERN'*)

run()
Run the thread.

Arguments: None

Return: None

run_auto3d(*root_name*)
Run AutoSPHIRE. In case of Feedback rounds, just work with what is available. Otherwise, first wait until the required minimum number of classes is reached. Once this condition is met, create the combined classes file and provide it to the first AutoSPHIRE run. Afterwards, when the root_name is None start AutoSPHIRE when the minimum number of particles is met.

root_name - name of the file to process.

Returns: None

run_class2d(*root_name*)

run_command(*command*, *log_prefix*, *block_gpu*, *gpu_list*, *shell*, *file_to_delete=None*, *root_name_input='INVALID'*)
Run the command with respect to the gpu list.

command - Command to run block_gpu - Block the GPU gpu_list - List of GPUs to use settings - Transphire settings

Return: log_file name, err_file name

run_compress(*root_name*)
Compress stack.

root_name - Name of the file to compress

Returns: None

run_copy_extern(*root_name*)
Copy to Work/Backup/HDD

root_name - Root name of the file to copy

Returns: None

run_ctf(*root_name*)
Run CTF estimation.

root_name - name of the file to process.

Returns: None

run_extract(*root_name*)
Run Particle extraction.

root_name - name of the file to process.

Returns: None

run_find()
Find files

Arguments: None

Return: None

run_import(*root_name*)
Import found files to em-transfer.

root_name - Root name of file to copy.

Returns: None

run_motion(*root_name*)
Do the motion correction.

root_name - Root name of the micrograph.

Returns: None

run_picking(*root_name*)
Run picking particles.

root_name - name of the file to process.

Returns: None

run_select2d(*root_name*)
Run Particle extraction.

root_name - name of the file to process.

Returns: None

run_software_meta(*directory*)
Copy meta files produces by the collection software.

Arguments: directory - Start directory for recursive search.

Return: None

run_train2d(*root_name*)
Run Particle extraction.

root_name - name of the file to process.

Returns: None

send_out_of_range_error(*warning*, *file_name*, *error_type*)

start_queue(*clear_list*)
Start pipeline processes.

Arguments: None

Return: None

start_queue_find()
Start finding files to process.

Arguments: None

Return: None

start_queue_meta()
Start copying meta files.

Arguments: None

Return: None

static try_write(**args*, ***kwargs*)

wait(*wait_time=10*)

write_error(*msg*, *root_name*)
Write to error file.

Arguments: msg - Message to send. root_name - File that was processed while the error occured.

Return: None

9.8.26 transphire.processworker module

TranSPHIRE is supposed to help with the cryo-EM data collection Copyright (C) 2017 Markus Stabrin

This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program. If not, see <http://www.gnu.org/licenses/>.

class transphire.processworker.**MyManager**(*address=None*, *authkey=None*, *serializer='pickle'*, *ctx=None*)
Bases: multiprocessing.managers.BaseManager

LifoQueue(**args*, ***kwds*)

class transphire.processworker.**ProcessWorker**(*password*, *content_process*, *mount_directory*, *parent=None*)
Bases: PyQt5.QtCore.QObject

Setup and start worker threads

Inherits from: QObject

Buttons: None

Signals: sig_start - Connected to the run method to start the process (settings|object) sig_finished - Emitted, if run method finishes (No objects) sig_error - Emitted, if an error occured (text|str) sig_status - Emitted to change the status (text|str, device|str, color|str) sig_notification - Emitted to send a notification (text|str) sig_plot_ctf - Emitted to plot ctf information (ctf_name|str, ctf_settings|object, settings|object) sig_plot_motion - Emitted to plot motion information (motion_name|str, motion_settings|object, settings|object) sig_plot_picking - Emitted to plot picking information (picking_name|str, picking_settings|str, settings|object)

check_queue(*queue_com*)
Check the content of the queues and react accordingly.

Arguments: queue_com

Return: None

emit_plot_signals(*folder_list*, *monitor*)

fill_spot_dict()
Fill the spot dictionary.

Arguments: None

Return: Spot dictionary

pre_check_programs()
Check, if all programs the user wants to use are available.

Arguments: None

Return: True, if programs exist, else False

prefill_queue(*shared_dict*, *entry*, *restart_dict*, *keep_list*)
Prefill the queues for continue mode

Arguments: shared_dict - Shared dictionary entry - Name of the queue process

Return: None

run(*settings*, *restart_dict*)
Start the process.

Arguments: settings - Transphire settings

Return: None

static run_in_parallel(*thread_obj*)

run_monitor(*typ_dict*, *queue_com*, *full_content*)
Run the TranSPHIRE monitor process.

Arguments: typ_dict - Dictionary for the queue types queue_com - Dictionary for queue communication

Returns: None

run_process(*typ_dict*, *queue_com*, *share_dict*, *bad_dict*, *queue_dict*, *content_process*, *full_content*, *manager*, *restart_dict*)
Run the TranSPHIRE process.

Arguments: typ_dict - Dictionary for the queue types queue_com - Dictionary for queue communication

Returns: None

sig_error

sig_finished

sig_notification

sig_set_project_directory

sig_start

sig_status

signal_plot

9.8.27 transphire.separator module

class transphire.separator.**Separator**(*typ*, *color*, *up=False*, *down=False*, *left=False*, *right=False*, *parent=None*)

Bases: PyQt5.QtWidgets.QWidget

Separator widget

hide_show_widget(*status*)

Hide or show the widget with the status.

Arguments: status - Status show or status hide.

9.8.28 transphire.settingscontainer module

class transphire.settingscontainer.**SettingsContainer**(*content*, *name*, *global_dict*, *settings_folder*, *mount_worker*, *parent=None*, ***kwargs*)

Bases: PyQt5.QtWidgets.QWidget

Widget for setting widgets.

Inherits: QWidget

change_state(*name*)

Change the state of widgets based on the choice of the corresponding combo box

name - Name of the group to change status (Emitted by the combo box)

Returns: None

emit_global(*_=None*)

emit_signals()

enable(*var*, *use_all*)
Disable or enable all widgets

Arguments: var - If True, enable the widgets, else disable them use_all - If True, all widgets are enabled/disabled

Returns: None

get_input_names()

get_settings(*quiet=False*)
Get the settings as dict

Arguments: quiet - If True, no prints are executed

Returns: Settings as dictionary

prepare_send_adjust(*text*)

static recursive_search(*folder*, *match*, *matches*)

search_for_projects(*project_dir*)

set_design(*settings*)
Set settings to the widgets

Arguments: settings - Settings to set.

Returns: None

set_global(*global_dict*)

set_new_model(*weights*, *threshold*)

set_settings(*settings*)
Set settings to the widgets

Arguments: settings - Settings to set.

Returns: None

sig_adjust_tab

sig_change_use_movie

update_global(*text*)

9.8.29 transphire.settingswidget module

TranSPHIRE is supposed to help with the cryo-EM data collection Copyright (C) 2017 Markus Stabrin

This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program. If not, see <http://www.gnu.org/licenses/>.

class transphire.settingswidget.**SettingsWidget**(*name*, *content*, *content_others*, *mount_directory*, *global_dict=None*, *input_file_names=None*, *parent=None*)

Bases: PyQt5.QtWidgets.QWidget

Widget for setting entrys

Inherits: QWidget

Signals: sig_index_changed - Emitted, if the index of a combo box changes (Combo box name|str)

change_color_if_true()
Change the color, if the types are all true.

Arguments: None

Returns: None

change_tooltip(*text*)

enlarge()

get_combo_entries()
Get the entries of the combo boxes.

Arguments: None

Returns: List containing entries.

get_current_global()

get_settings(*quiet=False*)
Get the settings as dict.

Arguments: quiet - True, if prints should not be shown.

Returns: None, if an error occured. Settings as dictionary.

set_settings(*text*, *is_checked*)
Set settings

text - Text to set to the widget.

Returns: None

sig_index_changed

9.8.30 transphire.statuscontainer module

TranSPHIRE is supposed to help with the cryo-EM data collection Copyright (C) 2017 Markus Stabrin

This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program. If not, see <http://www.gnu.org/licenses/>.

class transphire.statuscontainer.**StatusContainer**(*content*, *content_mount*, *content_pipeline*, *mount_worker*, *process_worker*, *content_font*, *parent=None*, ***kwargs*)

Bases: PyQt5.QtWidgets.QWidget

Container for status widgets

Inherits: QWidget

Signals: sig_refresh_quota - Connected to change the quota text (no object)

refresh_quota(*text*, *device*, *color*)

Refresh the quota

Arguments: text - Text to show device - Device name color - Color of the text

Returns: None

sig_refresh_quota

9.8.31 transphire.statuswidget module

TranSPHIRE is supposed to help with the cryo-EM data collection Copyright (C) 2017 Markus Stabrin

This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program. If not, see <http://www.gnu.org/licenses/>.

class transphire.statuswidget.**StatusWidget**(*name*, *default_name*, *default_quota*, *parent=None*)

Bases: PyQt5.QtWidgets.QWidget

Widget to show the current status of the processes.

Inherits: QWidget

Signals: sig_change_info_name - Connected, to change the info name information (text|str, color|str) sig_change_info_quota - Connected, to change the quota name information (text|str, color|str)

change_info_name(*text*, *color*)

Change the info text and the info text color.

Arguments: text - Text to put color - Color to use

Returns: None

change_info_quota(*text*, *color*)

Change the info text and the info text color.

Arguments: text - Text to put color - Color to use

Returns: None

sig_change_info_name

sig_change_info_quota

9.8.32 transphire.tabdocker module

TranSPHIRE is supposed to help with the cryo-EM data collection Copyright (C) 2017 Markus Stabrin

This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program. If not, see <http://www.gnu.org/licenses/>.

class transphire.tabdocker.**MyTabBar**(*parent*)

Bases: PyQt5.QtWidgets.QTabBar

paintEvent(*self*, *a0: QPaintEvent*)

tabSizeHint(*self*, *index: int*) → QSize

class transphire.tabdocker.**TabDocker**(*parent=None*, ***kwargs*)

Bases: PyQt5.QtWidgets.QWidget

Tab widget for the settingswidgets.

Inherits: QWidget

add_tab(*widget*, *name*, *add_widgets=True*)

Add a new tab to the TabDocker

Arguments: widget - Widget to add name - Name of the widget

Return: None

assign_latest(*idx*)

count()

Return the number of tabs.

Arguments: None

Returns: Number of tabs

currentIndex()

Get the current Index of the tab_widget.

Returns: Current index of self.tab_widget

enable_tab(*visible*)

Enable or disable the tab.

Arguments: visible - Enable if True, Disable if False name - Name of the tab to disable.

Returns: None

indexOf(*widget*)

Get the index of the widget.

Arguments: widget - Adress of the widget

Returns: Index of the widget

latest_active = [None]

order_tabs()

removeTab(*idx*)
Remove the widget located at tab idx

Arguments: idx - Idx of the widget

Returns: None

setCurrentIndex(*idx*)
Set the current Index of the tab_widget.

Arguments: idx - Index to set

Returns: Current index of self.tab_widget

setCurrentWidget(*widget*)
Set the current widget of the tab_widget.

Arguments: idx - Widget to set

Returns: Current index of self.tab_widget

setMovable(*status*)
Set the movable status for the tab widgets

Arguments: status - Boolean variable for the status

Returns: None

setTabEnabled(*index*, *state*)
Set the tab position index to the enable state.

Arguments: index - Tab position index state - State (True or False)

Returns: None

setTabPosition(*position*)
Set the tab position of the Tab bar

Arguments: position - Tab position as string [‘North’, ‘East’, ‘West’, ‘South’]

Returns: None

setTabText(*idx*, *text*)
Set the text for the tab at idx

Arguments: idx - Index of the tab text - Text of the tab

Returns: None

sig_start_plot

tabText(*idx*)
Return the text of the tab at idx

Arguments: idx - Index of the tab

Returns: Text of the tab at position isx

widget(*idx*)
Return the widget that belongs to the idx of tabs.

Arguments: idx - Tab index

Returns: Widget

9.8.33 transphire.templatedialog module

class transphire.templatedialog.**TemplateDialog**(*settings_directory*, *add_remove=True*, *parent=None*)

Bases: PyQt5.QtWidgets.QDialog

TemplateDialog dialog. Dialog used to enter the template.

Inherits from: QDialog

add_template()

choose_template()

copy_template()

remove_template()

9.8.34 transphire.transphire_class2d module

TranSPHIRE is supposed to help with the cryo-EM data collection Copyright (C) 2017 Markus Stabrin

This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program. If not, see <http://www.gnu.org/licenses/>.

transphire.transphire_class2d.**create_class2d_command**(*class2d_name*, *stack_name*, *file_name*, *output_dir*, *settings*, *queue_com*, *name*)

Create the command to combine BDB stacks.

transphire.transphire_class2d.**create_isac2_1_2_combine_command**(*class2d_name*, *file_names*, *file_name*, *output_dir*, *settings*)

Create e2bdb.py combine command

transphire.transphire_class2d.**create_isac2_1_2_command**(*class2d_name*, *stack_name*, *file_name*, *output_dir*, *settings*, *name*)

Create e2bdb.py combine command

transphire.transphire_class2d.**create_stack_combine_command**(*class2d_name*, *file_names*, *file_name*, *output_dir*, *settings*, *queue_com*, *name*)

Create the command to combine BDB stacks.

transphire.transphire_class2d.**find_logfiles**(*root_path*, *settings*, *queue_com*, *name*)

Find logfiles related to the produced Extract files.

root_path - Root path of the file settings - TranSPHIRE settings queue_com - Queue for communication name - Name of process

Returns: list of log files

transphire.transphire_class2d.**recursive_file_search**(*directory*, *files*)
Recursive file search function.

9.8.35 transphire.transphire_content module

TranSPHIRE is supposed to help with the cryo-EM data collection Copyright (C) 2017 Markus Stabrin

This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program. If not, see <http://www.gnu.org/licenses/>.

transphire.transphire_content.**default_auto_sphire_v1_3**()
Content for auto sphire

Arguments: None

Return: Content items as list

transphire.transphire_content.**default_cinderella_v0_3_1**()
Content of cinderella v0.3.1

Arguments: None

Return: Content items as list

transphire.transphire_content.**default_compress_command_line**()
Content for compression

Arguments: None

Return: Content items as list

transphire.transphire_content.**default_copy**(*settings_folder*)
Content of Copy tab.

Arguments: None

Return: Content items as list

transphire.transphire_content.**default_cryolo_train_v1_5_4**()
Content of crYOLO_train version 1.5.4

Arguments: None

Return: Content items as list

transphire.transphire_content.**default_cryolo_train_v1_5_8**()
Content of crYOLO_train version 1.5.8

Arguments: None

Return: Content items as list

transphire.transphire_content.**default_cryolo_train_v1_7_4**()
Content of crYOLO_train version 1.7.4

Arguments: None

Return: Content items as list

transphire.transphire_content.**default_cryolo_v1_0_4**()
Content of crYOLO version 1.0.4

Arguments: None

Return: Content items as list

transphire.transphire_content.**default_cryolo_v1_1_0**()
Content of crYOLO version 1.1.0

Arguments: None

Return: Content items as list

transphire.transphire_content.**default_cryolo_v1_2_1**()
Content of crYOLO version 1.2.1

Arguments: None

Return: Content items as list

transphire.transphire_content.**default_cryolo_v1_4_1**()
Content of crYOLO version 1.4.1

Arguments: None

Return: Content items as list

transphire.transphire_content.**default_cryolo_v1_5_8**()
Content of crYOLO version 1.5.8

Arguments: None

Return: Content items as list

transphire.transphire_content.**default_cter_v1_0**()
Content of CTER SPHIRE version 1.0.

Arguments: None

Return: Content items as list

transphire.transphire_content.**default_ctffind_4_v4_1_8**()
Content of CTFFind version 4.1.8.

Arguments: None

Return: Content items as list

transphire.transphire_content.**default_font**()
Content of Font tab.

Arguments: None

Return: Content items as list

transphire.transphire_content.**default_gctf_v1_06**()
Content of GCtf version 1.06.

Arguments: None

Return: Content items as list

transphire.transphire_content.**default_gctf_v1_18**()
Content of GCtf version 1.18.

Arguments: None

Return: Content items as list

transphire.transphire_content.**default_general**()
Content of General tab.

Arguments: None

Return: Content items as list

transphire.transphire_content.**default_global**()

transphire.transphire_content.**default_input**()
Content of General tab.

Arguments: None

Return: Content items as list

transphire.transphire_content.**default_isac2_1_2**()
Content of sp_isac2(gpu) version 1.2

Arguments: None

Return: Content items as list

transphire.transphire_content.**default_motion_cor_2_v1_0_0**()
Content of MotionCor2 version 1.0.0.

Arguments: None

Return: Content items as list

transphire.transphire_content.**default_motion_cor_2_v1_0_5**()
Content of MotionCor2 version 1.0.5.

Arguments: None

Return: Content items as list

transphire.transphire_content.**default_motion_cor_2_v1_1_0**()
Content of MotionCor2 version 1.1.0.

Arguments: None

Return: Content items as list

transphire.transphire_content.**default_motion_cor_2_v1_3_0**()
Content of MotionCor2 version 1.3.0.

Arguments: None

Return: Content items as list

transphire.transphire_content.**default_mount**(*hdd=None*)
Content of Mount tab.

Arguments: hdd - Content is related to HDD

Return: Content items as list

transphire.transphire_content.**default_notification**()
Content of notification tab.

Arguments: None

Return: Content items as list

transphire.transphire_content.**default_notification_widget**()
Content of notification widget.

Arguments: None

Return: Content items as list

transphire.transphire_content.**default_others**(*settings_folder*)
Content of Status tab.

Arguments: None

Return: Content items as list

transphire.transphire_content.**default_path**()
Content of Path tab.

Arguments: None

Return: Content items as list

transphire.transphire_content.**default_pipeline**()
Content of pipeline tab.

Arguments: None

Return: Content items as list

transphire.transphire_content.**default_unblur_v1_0_0**()
Content of Unblur version 1.0.0.

Arguments: None

Return: Content items as list

transphire.transphire_content.**default_window_1_2**()
Content of sp_window version 1.2

Arguments: None

Return: Content items as list

9.8.36 transphire.transphire_ctf module

transphire.transphire_ctf.**combine_ctf_outputs**(*data*, *data_orig*, *root_path*, *file_name*, *settings*, *queue_com*, *shared_dict*, *name*, *sum_file*, *dw_file*)

Combine the ctf outputs to one SPHIRE partres and one RELION star file.

root_path - Root path of the file file_name - File name of the ctf file. settings - TranSPHIRE settings queue_com - Queue for communication name - Name of process sum_file - Name of the dose uncorrected sum file

Returns: None

transphire.transphire_ctf.**contrast_to_shift**(*a_cont*)

Transform amplitude contrast to phase shift.

a_cont - Amplitude contrast value in percent

Returns: Phase shift in degrees.

transphire.transphire_ctf.**create_cter_1_0_command**(*ctf_name*, *file_sum*, *file_input*, *output_dir*, *settings*)

Create the CTER v1.0 command

transphire.transphire_ctf.**create_cter_1_3_command**(*ctf_name*, *file_sum*, *file_input*, *output_dir*, *settings*)

Create the CTER v1.3 command

transphire.transphire_ctf.**create_ctffind_4_v4_1_13_command**(*ctf_name*, *file_sum*, *file_input*, *file_output*, *set_name*, *settings*)

Create the ctffind command

transphire.transphire_ctf.**create_ctffind_4_v4_1_8_command**(*ctf_name*, *file_sum*, *file_input*, *file_output*, *set_name*, *settings*)

Create the ctffind command

transphire.transphire_ctf.**create_export_data**(*export_data*, *lines*, *maximum_string*)

Write export data to file.

export_data - Data to export. file_name - Name of the file to write to.

Returns: In place modificaion of lines

transphire.transphire_ctf.**create_gctf_v1_06_command**(*ctf_name*, *file_sum*, *file_input*, *file_output*, *settings*, *name*)

Create the Gctf v1.06 command

transphire.transphire_ctf.**find_logfiles**(*root_path*, *file_name*, *settings*, *queue_com*, *name*)

Find logfiles related to the produced CTF files.

root_path - Root path of the file file_name - File name of the ctf file. settings - TranSPHIRE settings queue_com - Queue for communication name - Name of process

Returns: list of log files

transphire.transphire_ctf.**get_constant_value**(*key*, *ctf_settings*, *row*, *project_folder*, *ctf_name*, *ctf_folder*, *pixel_size_adjust*)

transphire.transphire_ctf.**get_ctf_command**(*file_sum*, *file_input*, *new_name*, *settings*, *queue_com*, *set_name*, *name*)

Create the ctf command based on the ctf software.

file_input - Input name of the file for ctf estimation new_name - Output file settings - TranSPHIRE settings queue_com - Queue for communication name - Name of process

Returns: CTF command File to check vor validation if the process was successful

transphire.transphire_ctf.**get_relion_header**(*names*)
Create a relion star file header.

names - Header names as list

Returns: header string

transphire.transphire_ctf.**recursive_file_search**(*directory*, *files*)
Recursive file search function.

transphire.transphire_ctf.**shift_to_contrast**(*phase_shift*)
Convert phase shift to amplitud contrast.

phase_shift - Phase shift in degrees.

Returns: Amplitude contrast in percent.

transphire.transphire_ctf.**to_partres_file**(*data*, *ctf_name*, *ctf_settings*, *project_folder*, *ctf_folder*, *sum_file*, *pixel_size_adjust*)
Create a CTF partres file from data

data - Array containing ctf information. ctf_name - Name of the ctf program. ctf_settings - Settings for this ctf estimation run. project_folder - Name of the project folder. ctf_folder - Name of the ctf output folder. sum_file - Name of the summed micrograph image

Returns: None

transphire.transphire_ctf.**to_star_file**(*data*, *ctf_name*, *ctf_settings*, *project_folder*, *ctf_folder*, *sum_file*, *dw_file*, *pixel_size_adjust*)
Create a CTF star file from data

data - Array containing ctf information. ctf_name - Name of the ctf program. ctf_settings - Settings for this ctf estimation run. project_folder - Name of the project folder. ctf_folder - Name of the ctf output folder. sum_file - Name of the sum file

Returns: None

9.8.37 transphire.transphire_extract module

TranSPHIRE is supposed to help with the cryo-EM data collection Copyright (C) 2017 Markus Stabrin

This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program. If not, see <http://www.gnu.org/licenses/>.

transphire.transphire_extract.**create_window_1_2_command**(*extract_name*, *file_sum*, *file_box*, *file_ctf*, *output_dir*, *settings*)
Create the WINDOW v1.2 command

transphire.transphire_extract.**find_logfiles**(*root_path*, *settings*, *queue_com*, *name*)
Find logfiles related to the produced Extract files.

root_path - Root path of the file settings - TranSPHIRE settings queue_com - Queue for communication name - Name of process

Returns: list of log files

transphire.transphire_extract.**get_extract_command**(*file_sum*, *file_box*, *file_ctf*, *output_dir*, *settings*, *queue_com*, *name*)

Create the ctf command based on the ctf software.

file_input - Input name of the file for ctf estimation new_name - Output file settings - TranSPHIRE settings queue_com - Queue for communication name - Name of process

Returns: CTF command File to check vor validation if the process was successful

transphire.transphire_extract.**get_particle_number**(*log_file*, *settings*, *queue_com*, *name*)

transphire.transphire_extract.**recursive_file_search**(*directory*, *files*)

Recursive file search function.

9.8.38 transphire.transphire_import module

TranSPHIRE is supposed to help with the cryo-EM data collection Copyright (C) 2017 Markus Stabrin

This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program. If not, see <http://www.gnu.org/licenses/>.

transphire.transphire_import.**dummy**(*name*, *name_no_feedback*, *settings*, *directory_name*, *import_name=''*, *send_data=None*)

transphire.transphire_import.**get_dtype_dict**()

Dtype of the data plot array.

Arguments: None

Return: Dtype dict

transphire.transphire_import.**get_dtype_import_dict**()

Dtype of the file to import.

Arguments: None

Return: Dtype dict

transphire.transphire_import.**get_header**(*input_file*)

Extract header information from a relion star file.

Arguments: input_file - Input star file

Return: Color string

transphire.transphire_import.**get_relion_dict**()

Translate relion star file information to dtype dict.

Arguments: None

Return: Dtype dict

transphire.transphire_import.**get_transphire_dict**()
Translate transphire ctf dict into relion star file information.

Arguments: None

Return: Dtype dict

transphire.transphire_import.**import_auto_sphire_v1_3**(*name*, *name_no_feedback*, *settings*, *directory_name*, *import_name=''*, *send_data=None*)
Import motion information for auto_sphire.py version 1.3

Arguments: name - Name of motion program directory_name - Name of the directory to search for files

Return: Imported data

transphire.transphire_import.**import_cinderella_v0_3_1**(*name*, *name_no_feedback*, *settings*, *directory_name*, *import_name=''*, *send_data=None*)

transphire.transphire_import.**import_cryolo_v1_0_4**(*name*, *name_no_feedback*, *settings*, *directory_name*, *import_name=''*, *send_data=None*, *sub_directory=None*)
Import picking information for crYOLO v1.0.4.

Arguments: name - Name of picking program directory_name - Name of the directory to search for files

Return: Imported data

transphire.transphire_import.**import_cryolo_v1_2_2**(*name*, *name_no_feedback*, *settings*, *directory_name*, *import_name=''*, *send_data=None*)
Import picking information for crYOLO v1.2.2.

Arguments: name - Name of picking program directory_name - Name of the directory to search for files

Return: Imported data

transphire.transphire_import.**import_cryolo_v1_8_0**(*name*, *name_no_feedback*, *settings*, *directory_name*, *import_name=''*, *send_data=None*, *sub_directory=None*)
Import picking information for crYOLO v1.8.0.

Arguments: name - Name of picking program directory_name - Name of the directory to search for files

Return: Imported data

transphire.transphire_import.**import_cter_v1_0**(*name*, *name_no_feedback*, *settings*, *directory_name*, *import_name=''*, *send_data=None*)
Import ctf information for CTER v1.0. Defocus in angstrom, phase shift in degree.

Arguments: name - Name of ctf program directory_name - Name of the directory to search for files

Return: Imported data

transphire.transphire_import.**import_ctffind_v4_1_8**(*name*, *name_no_feedback*, *settings*, *directory_name*, *import_name=''*, *send_data=None*)
Import ctf information for CTFFIND v4.1.8. Defocus in angstrom, phase shift in degree.

Arguments: name - Name of ctf program directory_name - Name of the directory to search for files

Return: Imported data

transphire.transphire_import.**import_gctf_v1_06**(*name*, *name_no_feedback*, *settings*, *directory_name*, *import_name=''*, *send_data=None*)

Import ctf information for Gctf v1.06. Defocus in angstrom, phase shift in degree.

Arguments: name - Name of ctf program directory_name - Name of the directory to search for files

Return: Imported data

transphire.transphire_import.**import_isac_v1_2**(*name*, *name_no_feedback*, *settings*, *directory_name*, *import_name=''*, *send_data=None*)

transphire.transphire_import.**import_motion_cor_2_v1_0_0**(*name*, *name_no_feedback*, *settings*, *directory_name*, *import_name=''*, *send_data=None*)

Import motion information for MotionCor2 v1.0.0.

Arguments: name - Name of motion program directory_name - Name of the directory to search for files

Return: Imported data

transphire.transphire_import.**import_unblur_v1_0_0**(*name*, *name_no_feedback*, *settings*, *directory_name*, *import_name=''*, *send_data=None*)

Import motion information for cisTEM Unblur v1.0.0.

Arguments: name - Name of motion program directory_name - Name of the directory to search for files

Return: Imported data

transphire.transphire_import.**import_window_v1_2**(*name*, *name_no_feedback*, *settings*, *directory_name*, *import_name=''*, *send_data=None*)

9.8.39 transphire.transphire_motion module

TranSPHIRE is supposed to help with the cryo-EM data collection Copyright (C) 2017 Markus Stabrin

This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program. If not, see <http://www.gnu.org/licenses/>.

transphire.transphire_motion.**combine_motion_outputs**(*data*, *data_original*, *settings*, *queue_com*, *shared_dict*, *name*, *log_file*, *sum_file*, *dw_file*, *stack_file*, *set_name*)

Combine the motion outputs to one micrograph and one relion star file.

root_path - Root path of the file file_name - File name of the ctf file. settings - TranSPHIRE settings queue_com - Queue for communication name - Name of process sum_file - Name of the dose uncorrected sum file

Returns: None

transphire.transphire_motion.**create_export_data**(*export_data*, *lines*)
Write export data to file.

export_data - Data to export. file_name - Name of the file to write to.

Returns: In place modificaion of lines

transphire.transphire_motion.**create_motion_cor_2_v1_0_0_command**(*motion_name*, *file_input*, *file_output*, *file_log*, *settings*, *queue_com*, *name*, *set_name*, *do_subsum*)
Create the MotionCor2 v1.0.0 command

file_input - Input file for motion correction. file_output - Output filename file_log - Logfile name settings - TranSPHIRE settings. queue_com - Queue for communication. name - Name of the process.

Returns: Command for MotionCor2 v1.0.0

transphire.transphire_motion.**create_sum_movie_command**(*motion_frames*, *file_input*, *file_output*, *file_shift*, *file_frc*, *settings*, *queue_com*, *name*)
Create the SumMovie command.

motion_frames - Sub frames settings dictionary file_input - File to sum. file_output - Output file name file_shift - Output shift file name file_frc - Output frc file name settings - TranSPHIRE settings queue_com - Queue for communication name - Name of the process

Returns: Command for SumMovie

transphire.transphire_motion.**create_sum_movie_v1_0_2_command**(*motion_frames*, *file_input*, *file_output*, *file_shift*, *file_frc*, *settings*, *queue_com*, *name*)
Create the SumMovie v1.0.2 command.

motion_frames - Sub frames settings dictionary file_input - File to sum. file_output - Output file name file_shift - Output shift file name file_frc - Output frc file name settings - TranSPHIRE settings queue_com - Queue for communication name - Name of the process

Returns: Command for Summovie v1.0.2

transphire.transphire_motion.**create_unblur_v1_0_0_command**(*motion_name*, *file_input*, *file_output*, *file_log*, *settings*, *queue_com*, *name*, *set_name*, *do_subsum*)
Create the SumMovie v1.0.2 command.

motion_frames - Sub frames settings dictionary file_input - File to sum. file_output - Output file name file_shift - Output shift file name file_frc - Output frc file name settings - TranSPHIRE settings queue_com - Queue for communication name - Name of the process

Returns: Command for Summovie v1.0.2

transphire.transphire_motion.**get_dw_file_name**(*output_transfer_scratch*, *file_name*, *settings*, *queue_com*, *name*)

Get the name of the dose weighted file directly after the program finished.

output_transfer - Name of the folder in the scratch directory. file_name - File name of the root_name path. settings - TranSPHIRE settings. queue_com - Queue for communication. name - Name of the process.

Returns: File path of the DW file.

transphire.transphire_motion.**get_dws_file_name**(*output_transfer_scratch*, *file_name*, *settings*, *queue_com*, *name*)

Get the name of the dose weighted file directly after the program finished.

output_transfer - Name of the folder in the scratch directory. file_name - File name of the root_name path. settings - TranSPHIRE settings. queue_com - Queue for communication. name - Name of the process.

Returns: File path of the DW file.

transphire.transphire_motion.**get_motion_command**(*file_input*, *file_output_scratch*, *file_log_scratch*, *settings*, *queue_com*, *name*, *set_name*, *do_subsum*)

Get the command for the selected motion software.

file_input - Input file for motion correction. file_output_scratch - Output file name file_log_scratch - Logfile path on the scratch disc settings - TranSPHIRE settings. queue_com - Queue for communication. name - Name of the process.

Returns: Motion command

transphire.transphire_motion.**get_motion_default**(*settings*, *motion_frames*, *queue_com*, *name*)

Set the default values for the motion correction software.

settings - TranSPHIRE settings. motion_frames - Sub frame settings. queue_com - Queue for communication. name - Name of the process.

Returns: True, if dose weighting will be applied.

transphire.transphire_motion.**get_relion_header**(*names*)

Create a relion star file header.

names - Header names as list

Returns: header string

9.8.40 transphire.transphire_picking module

TranSPHIRE is supposed to help with the cryo-EM data collection Copyright (C) 2017 Markus Stabrin

This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program. If not, see <http://www.gnu.org/licenses/>.

transphire.transphire_picking.**create_box**(*jpg_data*, *maskcenters*, *box_size*)

transphire.transphire_picking.**create_circle**(*jpg_data*, *maskcenters*, *radius*)
Create a circle in a numpy array

Arguments: jpg_data - Data that needs to be changed maskcenters - center of mask radius - Radius of the circle

Returns: None, jpg_data will be changed in-place

transphire.transphire_picking.**create_cryolo_v1_0_4_command**(*picking_name*, *file_input*, *file_output*, *settings*, *name*)
Create the crYOLO v1.0.4 command

transphire.transphire_picking.**create_cryolo_v1_1_0_command**(*picking_name*, *file_input*, *file_output*, *settings*, *name*)
Create the crYOLO v1.1.0 command

transphire.transphire_picking.**create_cryolo_v1_4_1_command**(*picking_name*, *file_input*, *file_output*, *settings*, *name*)
Create the crYOLO v1.1.0 command

transphire.transphire_picking.**create_filter_command**(*file_input*, *settings*)

transphire.transphire_picking.**find_logfiles**(*root_path*, *file_name*, *settings*, *queue_com*, *name*)
Find logfiles related to the produced CTF files.

root_path - Root path of the file file_name - File name of the ctf file. settings - TranSPHIRE settings queue_com - Queue for communication name - Name of process

Returns: list of log files

transphire.transphire_picking.**get_picking_command**(*file_input*, *new_name*, *settings*, *queue_com*, *name*)
Create the picking command based on the picking software.

file_input - Input name of the file for ctf estimation new_name - Output file settings - TranSPHIRE settings queue_com - Queue for communication name - Name of process

Returns: Picking command File to check vor validation if the process was successful

9.8.41 transphire.transphire_plot module

TranSPHIRE is supposed to help with the cryo-EM data collection Copyright (C) 2017 Markus Stabrin

This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program. If not, see <http://www.gnu.org/licenses/>.

transphire.transphire_plot.**dummy**(*data*, *settings*, *label*)

transphire.transphire_plot.**get_mic_number**(*array*, *settings*, *as_int=True*)
Identify the micrograph number out of the name string.

Arguments: array - Array containing information settings - User provided settings

Return: Array of micrograph numbers

transphire.transphire_plot.**update_batch**(*data*, *settings*, *label*)
Update the plot for crYOLO v1.0.4.

Arguments: data - Data to plot settings - User provided settings label - Label of the plot

Return: x values, y values, label, title

transphire.transphire_plot.**update_cryolo_v1_0_4**(*data*, *settings*, *label*)
Update the plot for crYOLO v1.0.4.

Arguments: data - Data to plot settings - User provided settings label - Label of the plot

Return: x values, y values, label, title

transphire.transphire_plot.**update_ctf**(*data*, *settings*, *label*)
Update the plot for CTFFIND v4.1.8.

Arguments: data - Data to plot settings - User provided settings label - Label of the plot

Return: x values, y values, label, title

transphire.transphire_plot.**update_micrograph**(*data*, *settings*, *label*)
Update the plot for crYOLO v1.0.4.

Arguments: data - Data to plot settings - User provided settings label - Label of the plot

Return: x values, y values, label, title

transphire.transphire_plot.**update_motion**(*data*, *settings*, *label*)
Update the plot for MotionCor2 v1.0.0.

Arguments: data - Data to plot settings - User provided settings label - Label of the plot

Return: x values, y values, label, title

9.8.42 transphire.transphire_select2d module

TranSPHIRE is supposed to help with the cryo-EM data collection Copyright (C) 2017 Markus Stabrin

This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program. If not, see <http://www.gnu.org/licenses/>.

transphire.transphire_select2d.**create_cinderella_0_3_1_command**(*prog_name*, *file_input*, *file_output*, *settings*, *name*)
Create the Cinderella >=v0.3.1 command

transphire.transphire_select2d.**find_logfiles**(*root_path*, *settings*, *queue_com*, *name*)

transphire.transphire_select2d.**get_select2d_command**(*file_input*, *output_dir*, *settings*, *queue_com*, *name*)
Create the picking command based on the picking software.

file_input - Input name of the file for ctf estimation output_dir - Output file settings - TranSPHIRE settings queue_com - Queue for communication name - Name of process

Returns: Picking command File to check vor validation if the process was successful

9.8.43 transphire.transphire_software module

TranSPHIRE is supposed to help with the cryo-EM data collection Copyright (C) 2017 Markus Stabrin

This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program. If not, see <http://www.gnu.org/licenses/>.

transphire.transphire_software.**check_for_outlier**(*dict_name*, *data*, *file_name*, *settings*)

transphire.transphire_software.**check_nr_frames**(*frames*, *settings*, *force=False*)

Check if the nr of frames of the stack match the given nr of frames

Arguments: frames - List of found frames settings - TranSPHIRE settings

transphire.transphire_software.**check_outputs**(*zero_list*, *non_zero_list*, *exists_list*, *folder*, *command*)

Check, if the output files are present and have the proper size.

zero_list - List of files that should be zero non_zero_list - List of files that should not be zero

Returns: None

transphire.transphire_software.**extract_time_and_grid_information**(*root_name*, *settings*, *queue_com*, *name*)

Extract the time and grid information from the micrograph names.

Arguments: root_name - Name to extract information from. settings - TranSPHIRE settings. queue_com - Queue for communication. name - Name of the process.

Returns: hole_number, spot1_number, spot2_number, date, time

transphire.transphire_software.**find_all_files**(*root_name*, *compare_name_frames*, *compare_name_meta*, *settings*, *queue_com*, *name*)

Find other files that relate to root_name.

root_name - Root name of files to find. compare_name_meta - Name of the meta data to find. queue_com - Queue for communication. name - Name of the process

Returns: list of files related to root_name.

transphire.transphire_software.**find_frames**(*frames_root*, *compare_name*, *settings*, *queue_com*, *name*, *write_error*)

Find frames based on Software, Type and camera used.

frames_root - Root name of the frames. compare_name - Name to compare jpg and frames as time might differ. settings - TranSPHIRE settings used. queue_com - Queue for communication name - Name of process write_error - Write error function

Returns: None if the number of frames does not match user input. False if an error occured and the file needs to be skipped. True if the function was successful.

transphire.transphire_software.**find_related_frames_to_jpg**(*frames_root*, *root_name*, *settings*, *queue_com*, *name*)

Find related frames to the jpg file based on the used software.

Arguments: frames_root - Root name to search for related files root_name - Root name of the jpg file settings - TranSPHIRE settings queue_com - Queue for communication name - Name of the process

transphire.transphire_software.**get_copy_command_for_frames**(*settings*, *queue_com*, *name*)

Copy the frames based on stack or frames type.

settings - TranSPHIRE settings. queue_com - Queue for communication. name - Name of the process.

Returns: Command to use for copy.

transphire.transphire_software.**get_logfiles**(*log_prefix*)

Return the names of the log_file and error file.

Arguments: log_prefix - Prefix to use.

Returns: Name of log file, Name of error file

transphire.transphire_software.**get_x_dim**(*frames*, *settings*)

9.8.44 transphire.transphire_train2d module

TranSPHIRE is supposed to help with the cryo-EM data collection Copyright (C) 2017 Markus Stabrin

This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program. If not, see <http://www.gnu.org/licenses/>.

transphire.transphire_train2d.**create_eval_command**(*config_file*, *weight_file*, *log_file*, *settings*, *name*)

transphire.transphire_train2d.**create_restack_command**(*stack_name*, *output_dir*, *settings*)

transphire.transphire_train2d.**create_substack_command**(*class_average_name*, *input_stack*, *isac_dir*, *output_dir*, *settings*)

transphire.transphire_train2d.**create_train_command**(*sum_folder*, *box_folder*, *output_dir*, *name*, *settings*)

9.8.45 transphire.transphire_utils module

TranSPHIRE is supposed to help with the cryo-EM data collection Copyright (C) 2017 Markus Stabrin

This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program. If not, see <http://www.gnu.org/licenses/>.

class transphire.transphire_utils.**DataFrame**(*manager*, *file_path*)

Bases: object

append_values(**args*, ***kwargs*)

get_df(**args*, ***kwargs*)

get_index_where(**args*, ***kwargs*)

get_values(**args*, ***kwargs*)

load_df(**args*, ***kwargs*)

save_df(**args*, ***kwargs*)

set_df(**args*, ***kwargs*)

set_values(**args*, ***kwargs*)

value_in_column(**args*, ***kwargs*)

transphire.transphire_utils.**check_instance**(*value*, *typ*)

Check typ of value

Arguments: value - Value to check typ - Type to check

Return: Bool

transphire.transphire_utils.**copy**(*file_in*, *file_out*)

Copy file_in to a new location.

Arguments: file_in - Input file file_out - Output file

Return: None

transphire.transphire_utils.**copytree**(*root_src_dir*, *root_dst_dir*)

transphire.transphire_utils.**create_log**(**args*)

Add a time string to print statement.

args - Args to print

Returns: String with added time stemp.

transphire.transphire_utils.**find_best_match**(*name*, *dictionary*)

Find the best matching version key in a dictionary. Raises an assertion error it the return_key is not valid.

name - Name of the current version. dictionary - Dictionary to find the best matching version.

Returns: The best matching version Key.

transphire.transphire_utils.**find_latest_version**(*name*, *dictionary*)

Find the latest matching version key in a dictionary. Raises an assertion error it the return_key is not valid.

name - Name of the program dictionary - Dictionary to find the best matching version.

Returns: The best matching version Key.

transphire.transphire_utils.**get_color**(*typ*)
Color defined for the type

Arguments: typ - Typ of color

Return: Color string

transphire.transphire_utils.**get_content_gui**(*content*, *template_name*, *n_feedbacks*)
Create content lists to load the GUI.

Arguments: content - Content as dictionary. template_name = Name of the template!

Return: Content as list

transphire.transphire_utils.**get_exclude_set_path**(*content*)
Check the widget_2 variable, if the program should be loaded or not.

Argument: content - Content as dictionary.

Return: List of names to exclude

transphire.transphire_utils.**get_function_dict**()
Return a dictionary containing the function to use for specific plots.

Arguments: None

Return: None

transphire.transphire_utils.**get_key_names**(*settings_folder*, *name*)
Extract mount names from related settings file.

Arguments: None

Return: List of mount names

transphire.transphire_utils.**get_name**(*name*)
Remove the extension of a file and return the basename without the PATH

name - Name to remove the extension from

Returns: The name of the file without PATH and extension.

transphire.transphire_utils.**get_style**(*typ*)
Style colores for the content of widgets.

Arguments: typ - Typ of color

Return: Color string

transphire.transphire_utils.**get_unique_types**()

transphire.transphire_utils.**is_higher_version**(*name*, *version_ref*)
Compare the versions of software. The versions do not need to match in digits. v1 > v0, v0.0, v0.0.0 v1 >= v1 v1 < v1.0.1

name - Name of the Software containing the version string, e.g. v1, v1.1, v1.1.1, v1.1.1rc2 version_ref - reference version as string, e.g. 1, 1.1, 1.1.1

Returns: True if the version is larger than the reference version

transphire.transphire_utils.**look_and_feel**(*app*, *font=None*, *adjust_width=None*, *adjust_height=None*, *default=None*)
Look and feel.

Arguments: app - QApplication. font - User provided font size (default None) adjust_width - User provided width adjustment (default None) adjust_height - User provided height adjustment (default None) default - Default values (default None)

Return: Style sheet

transphire.transphire_utils.**look_and_feel_small**(*app*, *font=None*)
Look and feel for the default settings dialog.

Arguments: app - QApplication. font - User provided font size (default None)

Return: Style sheet

transphire.transphire_utils.**message**(*text*)
Show a text in a message box.

Arguments: text - Text shown in the message box

Return: None

transphire.transphire_utils.**mkdir_p**(*path*)
Create output directories recursively.

Arguments: path - Directory path to create

Return: None

transphire.transphire_utils.**normalize_image**(*data*, *apix=1.0*, *min_res=30*, *real=True*)

transphire.transphire_utils.**question**(*head*, *text*)
Show a questions message box dialog.

Arguments: head - Header of the window text - Text with the questions

Return: True if No, False if Yes

transphire.transphire_utils.**rebin**(*arr*, *new_shape*)

transphire.transphire_utils.**reduce_copy_entries**(*exclude_set*, *content*)
Reduce the number of options based on the exclude_set

Arguments: exclude_set - Set of names to not consider content - Content of the widgets

Return: None

transphire.transphire_utils.**reduce_path_widget**(*exclude_set*, *content*)
Reduce the number of paths based on the exclude_set

Arguments: exclude_set - Set of names to not consider content - Content of the widgets

Return: None

transphire.transphire_utils.**reduce_programs**(*exclude_set=None*)
Reduce the number of programs to the users preferences.

Arguments: exclude_set - Set of names to not consider

Return: List of content for motion, List of content for ctf

transphire.transphire_utils.**rerun_function_in_case_of_error**(*func*)

transphire.transphire_utils.**split_maximum**(*text*, *max_char*, *split_char=None*)
Split text into chunks of size max_char containing whole words.

Arguments: text - Text to split max_char - Maximum number of characters

Returns: Splitted text

transphire.transphire_utils.**symlink_rel**(*src*, *dst*)

transphire.transphire_utils.**thread_safe**(*func*)

transphire.transphire_utils.**tooltip_style**()

9.8.46 Module contents

CHAPTER 10

Citation

Main paper: Stabrin, M., Schoenfeld, F., Wagner, T. et al. TranSPHIRE: automated and feedback-optimized on-the-fly processing for cryo-EM. Nat Commun 11, 5716 (2020). https://doi.org/10.1038/s41467-020-19513-2

Please also properly cite the individual tools that you used during the TranSPHIRE run.

CHAPTER 11

Indices and tables

- genindex
- modindex
- search

Python Module Index

t

Index

A

C

D

E

F

G

H

I

L

M

N

O

P

Q

R

S

T

U

V

W