LabelChecker - User manual

Version 1.0

Tim J. W. Walles, Katerina Symiakaki

January 2025

Table of contents

About

This manual introduces a workflow for processing FlowCam data using LabelChecker software. LabelChecker is open-source and designed to be easy-to-use and compatible with major operating systems such as, Windows, MacOS and Linux.

1.1 Workflow

The workflow starts during image data acquisition in VisualSpreadsheet®(Yokogawa Fluid Imaging Technologies, Inc., ME, USA), where **it is important to automatically export the CSV data files with the particle properties, so that they have the same name as the sample run. When setting the data export in VisualSpreadsheet, select all parameters so that no data is lost**. If the FlowCam data (image TIF collage files and CSV files with particle properties) are in order, the pipeline continues with:

- preprocessing - using a Python script, normalize the FlowCam data output and make it suitable for opening in LabelChecker, while removing artefacts from analysis, i.e. particles too small for classification, duplicate images and air bubbles

- validation of the preprocessing step

- classification - labelling imaged objects in user-defined categories (it can be taxonomic or other classification)

- (optional) validation of automatic classification

  If automatic classification models are used, the predicted labels can be assessed with the steps we describe, but *this manual does not cover training and predicting with a classification model.*

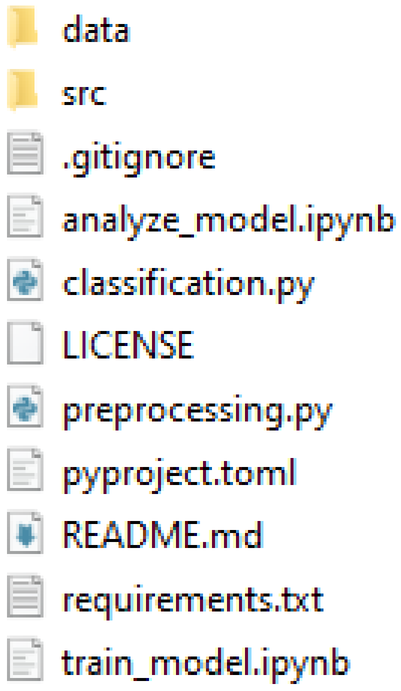
Download and installation

Python 3.12 is used to execute the preprocessing and classification steps within the data analysis pipeline. Please ensure that Python is installed.

2.1 Download scripts and tools

2.1.1 Data processing pipeline scripts

All scripts can be cloned or downloaded in a zip-format from LabelChecker data pipeline repository. You should have several files and folders (see fig. 2.1)



data
src
.gitignore
analyze_model.ipynb
classification.py
LICENSE
preprocessing.py
pyproject.toml
README.md
requirements.txt
train_model.ipynb

Figure 2.1: Documents and folders after unzipping

2.2 LabelChecker software

You can download the LabelChecker software for your operating system here and please follow the specific installation instructions below.

2.2.1 Windows

Unzip the file. Go into the folder and run the executable file.

> Note*: on Windows 11 a security window might pop up. Click "More" and then "Run Anyway" to open the program. You only need to do this action once.*

2.2.2 MacOS

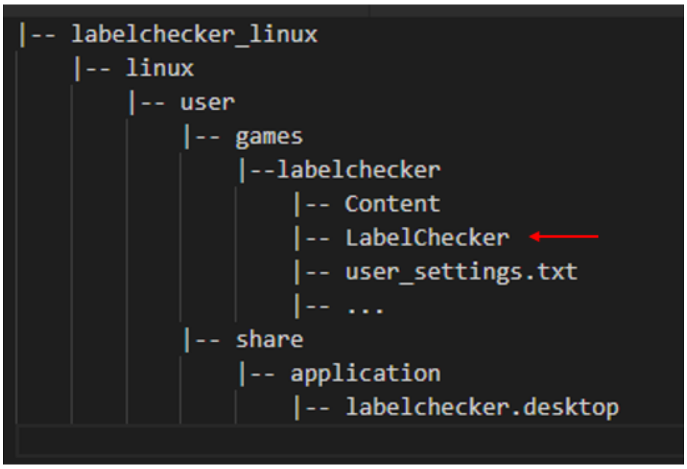
Unzip the file and run the executable file. You can drag the executable file to your application folder for easy access via the app-launcher button.

> Note*: a security window might pop up warning that the software is from an unknown developer. In that case, right click on the application and select open. This will trigger the pop-up warning again but you can now select "open anyway". You only need to do this action once. If the security warnings persist, follow the instructions here.*

If that doesn't work, please see Troubleshooting.

2.2.3 Linux

Unzip the file. LabelChecker will be in the directory as shown in fig. 2.2.



```
|-- labelchecker_linux
    |-- linux
        |-- user
            |-- games
                |--labelchecker
                    |-- Content
                    |-- LabelChecker ←
                    |-- user_settings.txt
                    |-- ...
            |-- share
                |-- application
                    |-- labelchecker.desktop
```

Figure 2.2: Folder structure of the Linux version of LabelChecker.

To have the software appear in your application, run the following command in the terminal:

```
sudo mkdir -p /usr/games/labelchecker
sudo mkdir -p /usr/share/applications

sudo cp -r /home/YourUserName/Downloads/LabelChecker-linux/linux/usr/games/LabelChecker/* /usr/games/LabelChecker/
sudo cp -r /home/YourUserName/Downloads/LabelChecker-linux/linux/usr/share/applications/LabelChecker.desktop /usr/share/applications

sudo chmod +x /usr/games/LabelChecker/LabelChecker
```

Preprocessing

In the preprocessing step, the names of the FlowCam CSV data get normalized, ensuring compatibility with LabelChecker. Additionally, it provides services to eliminate artefacts from the data, such as air bubbles and duplicate images, and set particle size thresholds. Each service operates independently, allowing users to customize or disable them as needed.

> *To run Python scripts, you'll need to install a few dependencies. We recommend creating a dedicated virtual environment for these, as it provides isolation and ensures that your project's packages won't interfere with other projects.*

3.1 Preprocessing script

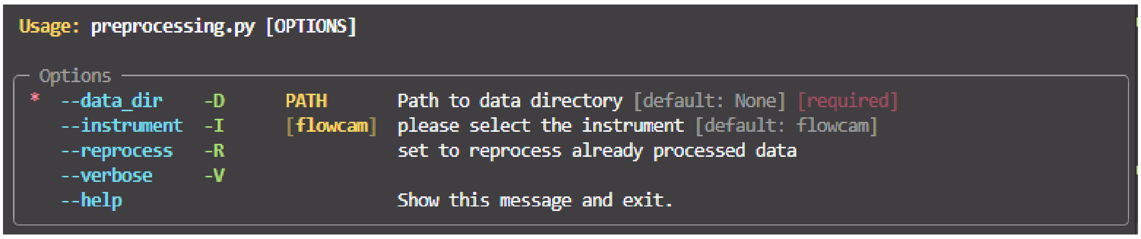
To preprocess the data you need to run the `preprocessing.py` script inside the virtual environment.

3.1.1 Helper function

To determine the required arguments for running the script, simply invoke the helper function by typing

```
python preprocessing.py --help
```

> Note: *if you receive an error (*"`ModuleNotFoundError: No module named <Module name>`"*) simply install the missing module with pip (type* pip install <Module name>*) and try the command again.*



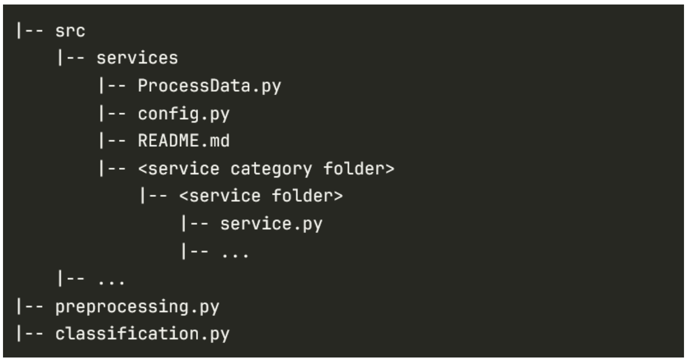
```
Usage: preprocessing.py [OPTIONS]

╭─ Options ─────────────────────────────────────────────────────────────────────────╮
│ * --data_dir     -D      PATH       Path to data directory [default: None] [required] │
│   --instrument   -I      [flowcam]  please select the instrument [default: flowcam]   │
│   --reprocess    -R                 set to reprocess already processed data           │
│   --verbose      -V                                                                   │
│   --help                            Show this message and exit.                       │
╰───────────────────────────────────────────────────────────────────────────────────╯
```

Figure 3.1: Helper output for `preprocessing.py`

With the helper function output you can see that several inputs (like the `-help` we just did) can be added to our command with the Python script. Furthermore, it shows information on what each input does and what the default setting is. Two inputs we must give is the `-D` or `-data_dir` input; the path to the image data.

3.1.2 Data-cleaning services

We have incorporated several services into the preprocessing step to detect and flag artefacts in the data, such as air bubbles and duplicates, and set an appropriate size threshold. You can configure each of these data-cleaning services by editing the `config.py` file located in the main services folder (fig. 3.2). In the `config.py` file you can enable or disable them (line 2: `active: bool = True/False`), or adjust their settings.



```
|-- src
    |-- services
        |-- ProcessData.py
        |-- config.py
        |-- README.md
        |-- <service category folder>
            |-- <service folder>
                |-- service.py
                |-- ...
    |-- ...
|-- preprocessing.py
|-- classification.py
```

Figure 3.2: Folder structure of the services, where the `config.py` file is

> Note*: The size threshold needs to be set on a size parameter for the object in the image, e.g. AbdDiameter, EsdDiameter or Length*

3.1.3 Preprocessing flowcam data

Let's preprocess the FlowCam data. As mentioned earlier, you need to provide the `-D` input: the directory containing the data output from the FlowCam.

> *Depending on your operating system, you might need to use \ instead of / when specifying the path.*

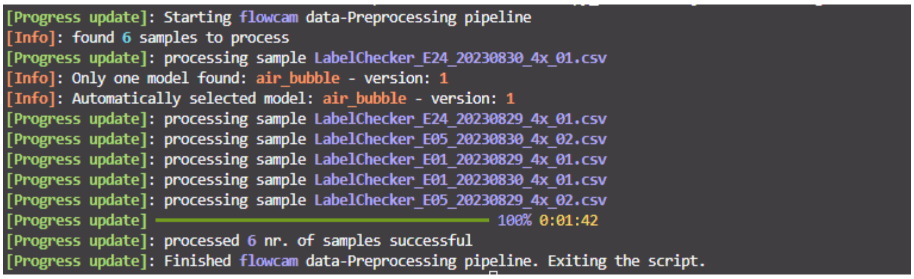
To execute the script, run the following command (note that everything else runs with default settings):

```
python preprocessing.py -D ./path/to/flowcam_data/
```

> *For clarification, the "." at the beginning of the path allows you to avoid writing the entire path starting from the root directory.*

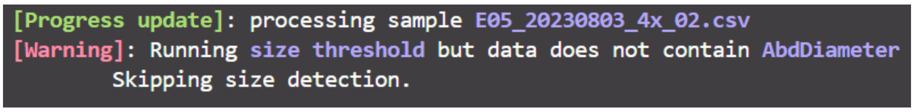
While running the script, you'll see a progress bar for each sample and an overall progress indicator. Larger data batches and samples with many pictures will take longer to process.

When we include the `-V` or `-verbose` option while running the command, the script provides additional information to the user. It informs us about the settings of each data-cleaning service, and during processing, it notifies us about the specific sample it's currently handling. If a service is turned off or the script found issue with the data, it warns us with relevant messages in output (see Fig. 3.4). These messages can help with debugging in case the script encounters errors or crashes.



```
[Progress update]: Starting flowcam data-Preprocessing pipeline
[Info]: found 6 samples to process
[Progress update]: processing sample LabelChecker_E24_20230830_4x_01.csv
[Info]: Only one model found: air_bubble - version: 1
[Info]: Automatically selected model: air_bubble - version: 1
[Progress update]: processing sample LabelChecker_E24_20230829_4x_01.csv
[Progress update]: processing sample LabelChecker_E05_20230830_4x_02.csv
[Progress update]: processing sample LabelChecker_E01_20230829_4x_01.csv
[Progress update]: processing sample LabelChecker_E01_20230830_4x_01.csv
[Progress update]: processing sample LabelChecker_E05_20230829_4x_02.csv
[Progress update] ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 100% 0:01:42
[Progress update]: processed 6 nr. of samples successful
[Progress update]: Finished flowcam data-Preprocessing pipeline. Exiting the script.
```

Figure 3.3: Output after running preprocessing.py



```
[Progress update]: processing sample E05_20230803_4x_02.csv
[Warning]: Running size threshold but data does not contain AbdDiameter
        Skipping size detection.
```

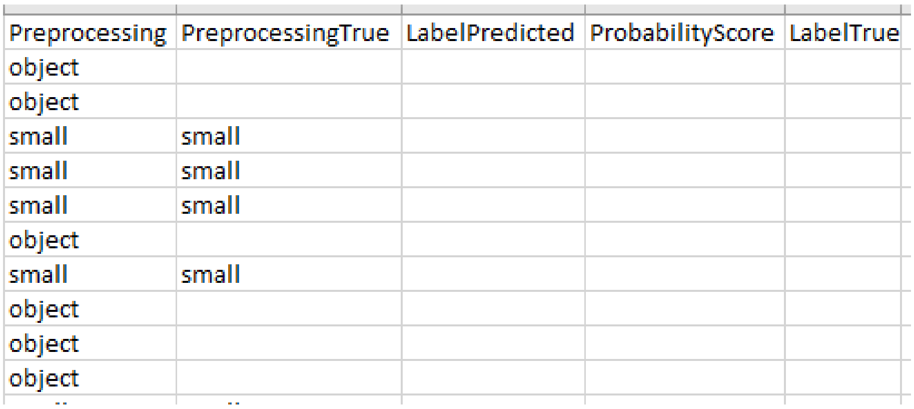
Figure 3.4: Warning messages when running the `preprocessing.py`

3.2 Script output

After running the `preprocessing.py` script, two new files are generated within each sample folder. One of them is a CSV file with the prefix "LabelChecker_" (hereafter LabelChecker_CSV), followed by the sample name, e.g. `LabelChecker_<sample code>.csv`. **It is crucial that this file remains inside the folder** so that the LabelChecker program can locate the images. The other file is called `processing_settings_summary.txt` and contains a copy of the preprocessing settings.

LabelChecker file

The preprocessing script generates LabelChecker_CSV files, which have content similar to the exported CSV data file produced by the FlowCam. A list of all the parameters can be found in LabelChecker file parameters table of the appendix. Additionally, the preprocessing script appends five columns at the end: `"Preprocessing"`, `"PreprocessingTrue"`, `"LabelPredicted"`, `"ProbabilityScore"` and `"LabelTrue"` (see Fig. 4.1). Any blank columns (without data) that are not present in the FlowCam CSV file are also included.



Preprocessing	PreprocessingTrue	LabelPredicted	ProbabilityScore	LabelTrue
object				
object				
small	small			
small	small			
small	small			
object				
small	small			
object				
object				
object				

Figure 4.1: Added columns at the end of the LabelChecler.csv file

- **Preprocessing**: Labels assigned to each object by the preprocessing script or its services. Possible labels include:
  - `object`: Objects of interest (default) within the size range.
  - `small`: Objects smaller than the given size threshold, excluded from analyses.
  - `large`: Objects larger than the given size threshold, excluded from analyses.
  - `duplicate`: Duplicated images, excluded from analyses.
  - `bubble`: Air bubbles (and calibration beads).
- **PreprocessingTrue**: Updated with the above labels after the user validates the preprocessing step. *Note* that small objects automatically get confirmed.
- **LabelPredicted**: Contains classification labels automatically predicted by a model (if used).
- **ProbabilityScore**: Contains the probability score of the predicted label if automatic classification is used.

- **LabelTrue**: Contains classification labels assigned by the user, either entirely or after validation of a classification model.

For how to perform the validation and classification steps see the Working with LabelChecker chapter.

The User Interface

To start the software, run the executable file of your operating system.

5.1 General layout

LabelChecker opens with a default layout that can be changed to the users liking. (fig. 5.1).

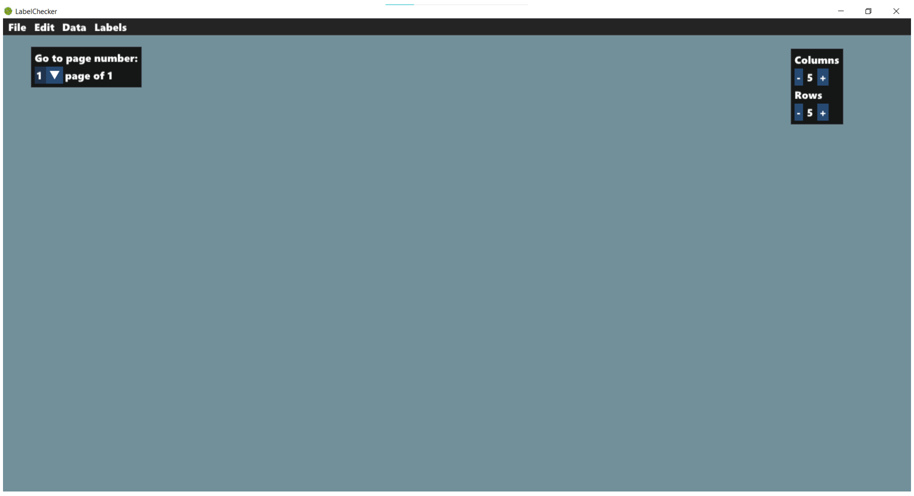


Figure 5.1: Default UI layout upon first time opening

As you open the software, two windows are on screen; a page number window and the image grid control (fig. 5.1). When a file is open you can see the total number of pages and go to a specific page. From the image grid control you can change the dimensions of the grid displaying the images - this allows you to adapt number of images, thereby their size and makes classification easier. Changing the grid size will also affect the number of pages.

Additionally, when there are already labelled images you can adjust the font size of the label (so that it fits in the grid) and filter out already checked images by toggling the respective boxes.

5.1.1 Modifying the UI layout

All windows can be moved, and except for the page number and image grid control, they can be resized according to the user's preference. To resize a window, click and drag the cursor from the bottom-right corner in any direction. Any layout changes made by the user are saved and restored when reopening LabelChecker.

5.2 Menu bar

The menu bar is found at the top left corner of the UI (see fig. 5.2). All option become available when a file is open.

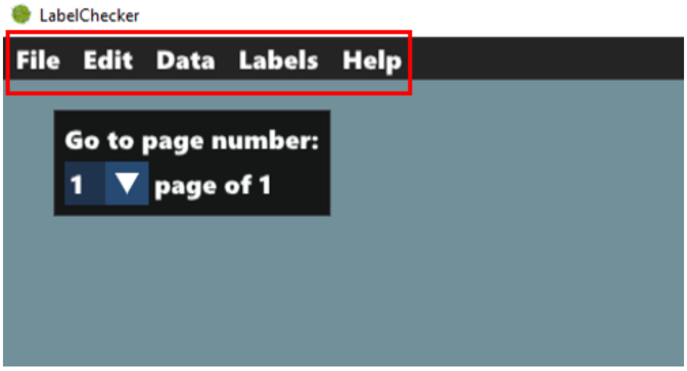


Figure 5.2: Menu bar

5.2.1 File

Under File menu you can:

- **Open file**: Opens a LabelChecker file; multiple files can also be selected (see chapter Working with LabelChecker).

- **Open folder**: Opens all or multiple LabelChecker files within a folder.

- **Open recent file**: Displays the 10 most recently opened LabelChecker files. Does not work for multiple files and folders.

- **Save**: Saves an open LabelChecker file.

- **Save as**: Saves an open LabelChecker file under a new name and/or location. Note that in the current LabelChecker version, saving under a new name resets the interface to the newly saved file. Use "Save as" either at the beginning or end of processing data in the file.

- **Settings**: Displays options for better user experience and visibility:
  - `Full screen`: Enables/disables full screen mode.
  - `Theme`: Choose among different themes.
  - `Auto size image labels`: Adjust the font size of the label of already labelled images so that it fits in the grid.
  - `Font size`: Customize the font size (affects all text in LabelChecker).
  - `Show file info`: Displays the Image ID, Image UUID and sample name when the cursor hovers over an image.
  - `Scale Images to fit`: Resize all small objects to match the cell size in the image grid.

  - `Magnification`: Show a magnified pop-up of the object when the cursor hovers over it.

- **Exit**: An alternative way to close the application.

5.2.2 Edit

- **Undo** last action
- **Redo** last action

5.2.3 Data

From this menu, you can customize how LabelChecker data is presented:

- **Correct label**: Allows you to rename a label within the `LabelChecker file` (active only when the Classification tab is selected; see chapter Working with LabelChecker).

- **Show data**:

  - `Checked only`: Display only the images that have a verified label (LabelTrue column is filled - shown with green border)
  - `Not checked only`: Display only the images that don't have a verified label (LabelTrue column is empty)

- **Show parameter value**: The user can select one of the FlowCam-measured parameters to display above each image. For an explanation of what each parameter is, please see LabelChecker file parameters

- **Order**: Choose between ascending (default) or descending order for data sorting.

- **Sort**: Sort images based on the following criteria:

  - `Id`: object id within the LabelChecker file - *corresponds to the order objects appear in the collage files* (default).
  - `AbdArea`
  - `AbdDiameter`
  - `AbdVolume`
  - `Length`
  - `Roughness`
  - `Transparency`
  - `Width`
  - `CircleFit`: Proximity of the object to a perfect circle.
  - `ProbabilityScore`: Probability score indicates how well the object fits the assigned taxonomical label.

- **Stats** `Labels confirmed`: Show the number of images in each class that have verified labels.

5.2.4 Labels

This menu allows you to interact with and edit Label files (see chapter: Label File).

- **New** - create a new Label file.
- **Open** - opens a Label file.
- **Open recent** - displays the 10 most recently opened Label files.
- **Save** - saves an open label file.
- **Save as** - saves an open Label file under a new name and/or location.
- **Add label** - add new label.
- **Rename label** - rename label.
- **Remove label** - remove label.

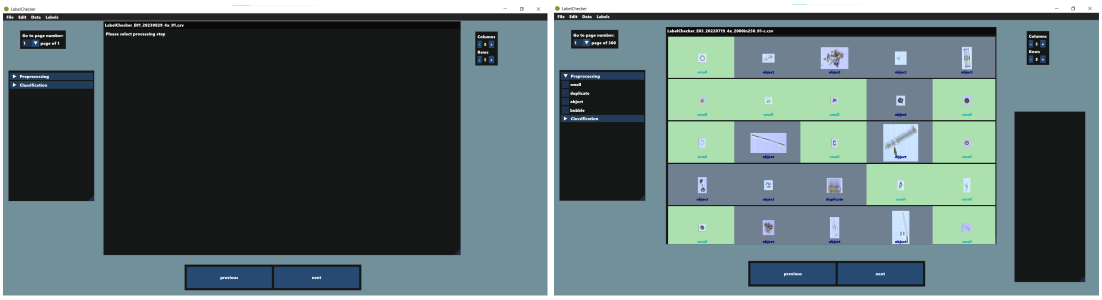
5.2.5 Help

- `About` - Information on LableChecker and license (GNU General Public License v3.0)
- `Support` - see

5.3 UI with an opened file

When you open a LabelChecker file more windows appear together with navigation buttons in the bottom (fig. 5.3).

- A window on the left with two tabs: "Preprocessing" and "Classification."
- In the middle, the image grid displays the opened file name. This grid displays images when the user selects either the "Preprocessing" or "Classification" tab.
- On the right, a window shows the labels and code. This window appears when a Label file (see chapter: Label File) is opened or when the "Classification" tab is selected.



LabelChecker_E01_20230829_4x_01.csv

Please select processing step

LabelChecker_E03_20220719_4x_2000in250_01-c.csv

Figure 5.3: Layout of windows when a file is opened and when a processing step is activated

5.3.1 Navigation

Navigating in the image grid can be done in several ways.

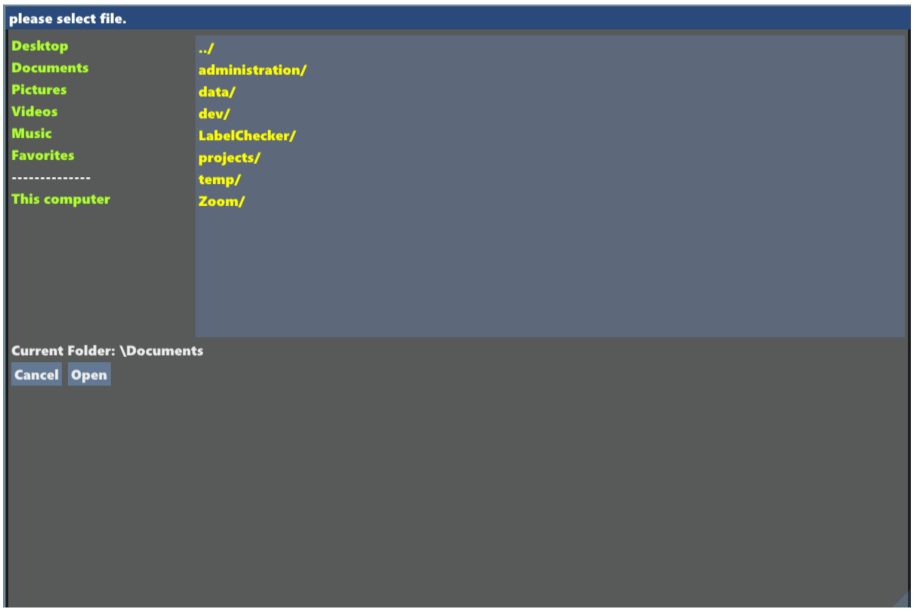
- by clicking the navigation buttons "next" and "previous" at the bottom.
- by selecting the page number you wish to see from the top left "Go to page number" window.
- by using the ← and → arrow keys.
- by using the ␣ spacebar to go to the next page.

Working with LabelChecker

LabelChecker uses two CSV files: a LabelChecker_CSV file (see chapter LabelChecker file) that is produced by the data preprocessing script, and a Label file that contains the labels and shortcut codes. The label file isn't a prerequisite and can be created using LabelChecker (see chapter Label File).

6.1 Open LabelChecker files

To open a LabelChecker file, select `‘‘File’’` –> `‘‘Open file’’` from the menu bar. A new pop-up window will appear (see Figure 6.1). By default, this window opens to the Documents folder. Navigate to the location of your LabelChecker file, select it, and click `Open`, or simply double-click the file to open it.



Current Folder: \Documents

Figure 6.1: Open file and select file pop-up

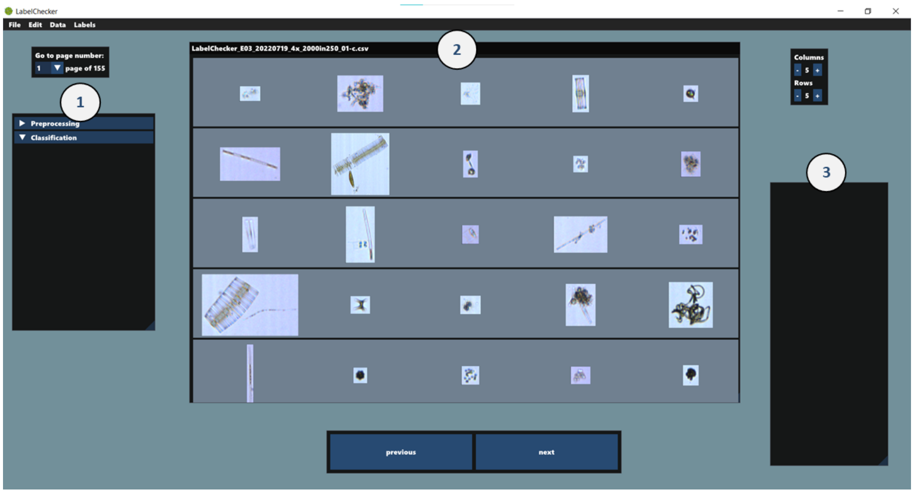
To work on multiple LabelChecker files simultaneously, select `‘‘File’’` –> `‘‘Open folder’’`. Instead of navigating to a single file, you can select a parent directory containing multiple LabelChecker files. You can also select which folders should be opened by holding down the ctrl (Windows, Linux) or ⌘ (MacOS) key, or with the ⇧ for consecutive folders.

> Note: *Opening many LabelChecker files at once can affect performance. The number of files that can be opened simultaneously without the program being slow partly depends on your hardware.*

A progress bar will appear showing the file(s) loading progress.

6.2 Processing steps

After opening a LabelChecker file the three windows: "Processing step", "Image grid" and "Navigation buttons" should appear (see fig. 5.3) When a processing step is selected an additional "Label file window" appears where the available labels will be displayed (fig. 6.2).



LabelChecker_E03_20220719_4x_2000in250_01-c.csv

Go to page number: 1 page of 155

Columns 5

Rows 5

Figure 6.2: Open windows when *Classification* is selected. 1: Processing step selection; 2: Image grid; 3: Label file window.

Following the Workflow, users can validate the automated output of the previous step (i.e. preprocessing or automatic classification) with LabelChecker by selecting the respective step, `"Preprocessing"` or `"Classification"`. Multiple LabelChecker files can be opened to speed-up the validation. Once one of the steps is selected, ***images with their original aspect ratios and relative proportions will appear on the grid, under the automatically assigned label. Once the label is verified, the image is moved to the correct class and the image border is turned green.***

> *When a user assigns/updates a label, the change is saved in the* `PreprocessingTrue` *or* `LabelTrue` *column when working on the preprocessed data or classified data, respectively.*

6.2.1 Preprocessing

When the user selects `"Preprocessing"` images appear in the image grid with the labels (fig. 6.3) assigned during the preprocessing step (see chapter: Preprocessing). The labels have different colours to ease identifying the different artefacts. Simultaneously, under `"Preprocessing"` checkboxes appear that can be checked to display only data of that label - multiple labels can be selected.

> Note*: checkboxes will appear of labels present in the data. For example, if no duplicates were found, no duplicate checkbox will appear until the user assigns that label to an object.*

> **Tip!** *When checking for duplicates, you can select both* duplicate *and* object*, and sort according to object Id. This way the images will appear in the order they were acquired and it will be easier to spot duplicate images, as they are usually close to each other.*

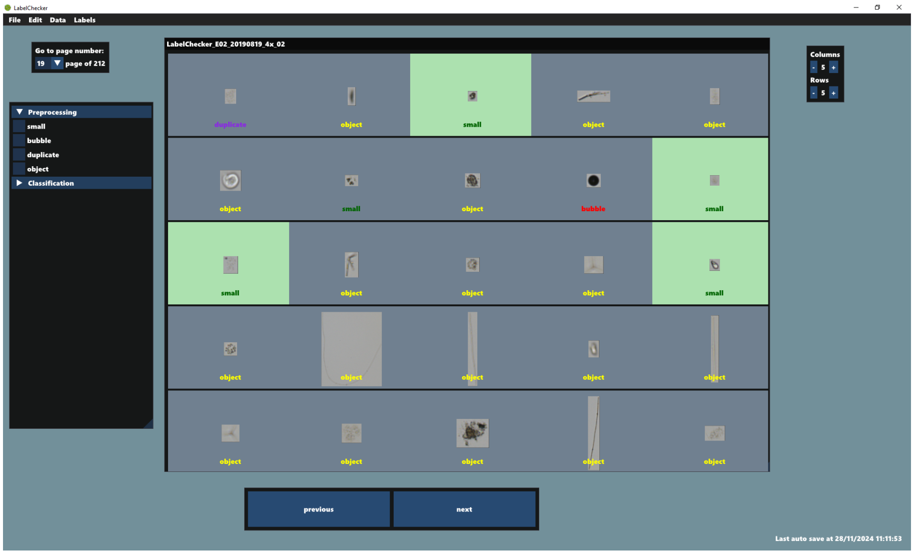


Figure 6.3: When preprocessing is selected, images are shown with different colour for the each label.

You can assign or correct labels to objects by selecting one (or more) image(s). A pop-up window will appear with label options. In case there is another type of artefact you would like to remove from the data, there is the `Add Label` option.

> Note: *When the size threshold service has been used during preprocessing, objects with the label* `"small"` *or* `"large"` *will already appear green because they get automatically validated by the service.*

6.2.2 Classification

When you select `"Classification"`, by default LabelChecker will initialize an empty label file window (fig: 6.3).

> Note: *the* Label file window *only exists for the duration of a session and* ***must*** *be saved if the users makes any changes to it within the software and wants to reuse it.*

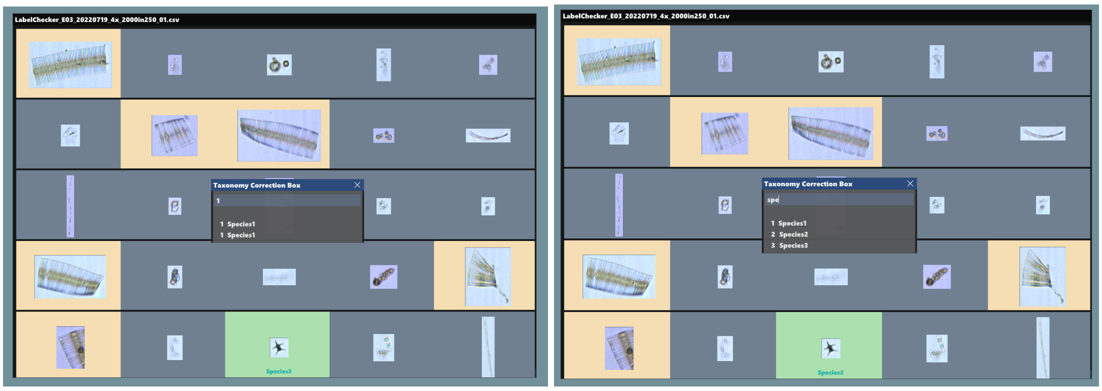
To classify your data, you need to either add your labels to the Label file window, or open a existing label file (`"Labels"` –> `"Open"`; see chapter Label File). The labels and their codes will appear in the window (see fig. 6.2).

Labelling/validating data

In the classification step users are either correcting automatically assigned labels or assigning them themselves. Corrections are saved in the LabelTrue column of the LabelChecker file(s), while assigned labels are saved both in the LabelPredicted and LabelTrue column.

After selecting one or more objects you want to label, a window appears, where you can enter either the label code or name (fig. 6.4). A list of labels and their code of suggestions appears as you type. You can assign the selected label by pressing ⏎ to assign that label to each object.

> Note: *The label shown at the top of the options list below the textbox gets assigned to the object(s).*


LabelChecker_E03_20220719_4x_2000in250_01.csv

Taxonomy Correction Box

1

1 Species1
1 Species1

LabelChecker_E03_20220719_4x_2000in250_01.csv

Taxonomy Correction Box

spe

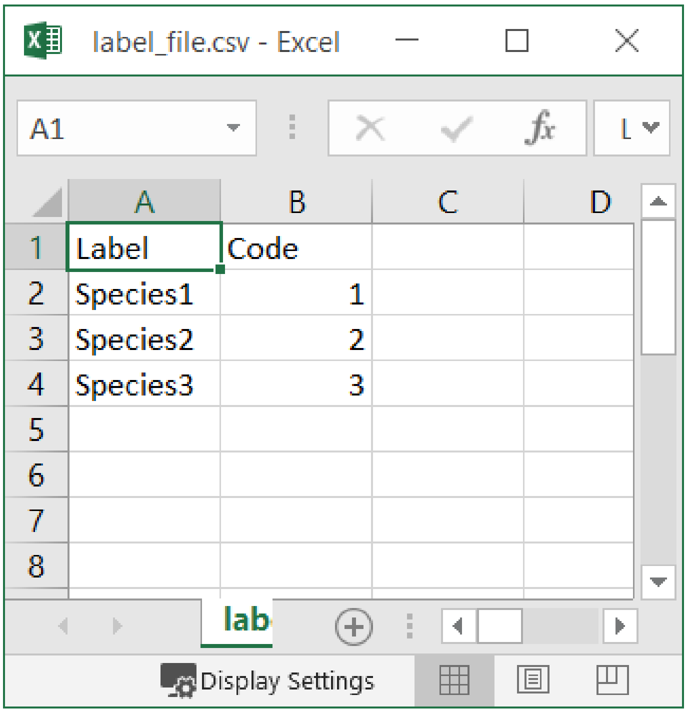
1 Species1
2 Species2
3 Species3

Figure 6.4: Labelling Species 1 either by typing the shortcut number or the name.

For a list of keyboard shortcuts you can use to speed up your work, see Keyboard shortcuts in the Appendix.

Label File

The Label file is a CSV file with two columns: `Label` and `Code`. Each row below the header contains the label name and corresponding code, as set by the user (see Fig. 7.1). Label files allow users to define different labels for specific projects. You can create a label file either outside or within LabelChecker, and we'll cover how to do the latter.



Label	Code
Species1	1
Species2	2
Species3	3

Figure 7.1: Example label file

7.1 Create a Label file

You can create a new Label file by selecting the menu item `"Labels"` -> `"New"`. When you haven't loaded in a Label file and select the "Classification" tab, LabelChecker will automatically initialize a Label file window.

> Note: *a* Label file window *only exists for the duration of a session and* **must** *be saved if the user wants to reuse it.*

7.2 Add label

You can add new labels by selecting to selecting menu `"Labels"` -> `"Add"`. A pop-up window appears (fig. 7.2) when you can type the label name and code. Once you click "add label", the label and code will appear in the label window on the right (fig. 7.3)

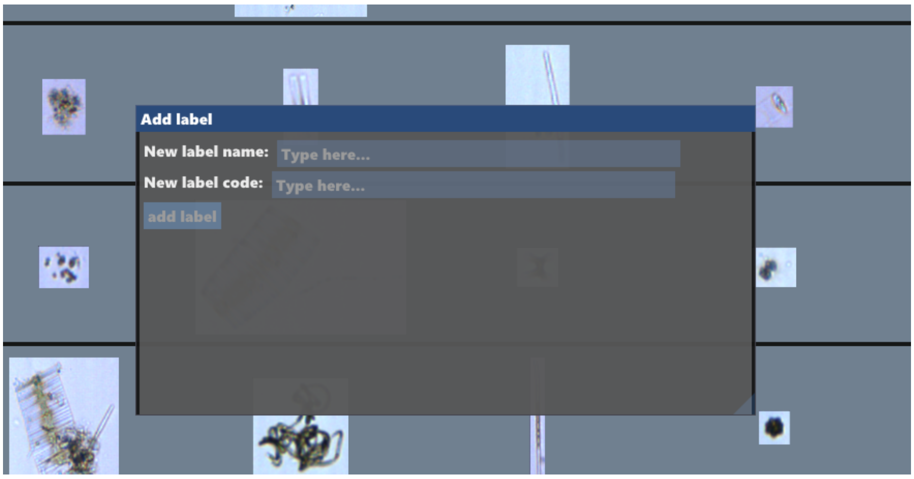
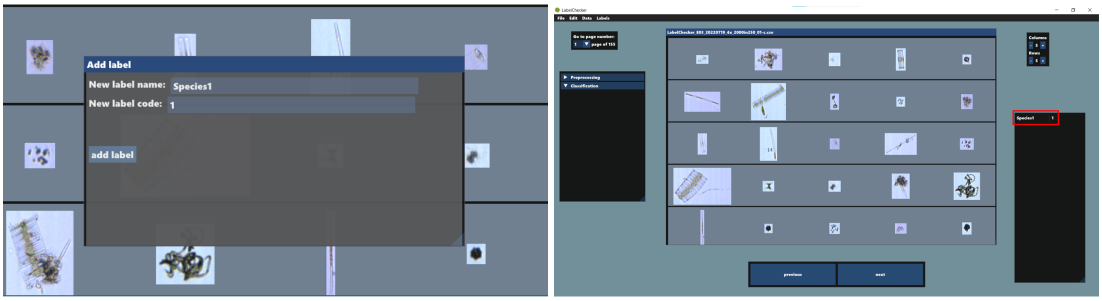


Figure 7.2: Add label box appears

> Note: *Each label and code* **must** *be unique. If not, the* "Add label" *window will give a message error text indicating the already existing label or code and won't allow you to add your label.*



New label name: Species1
New label code: 1

Species1 1

Figure 7.3: Adding Species 1

7.3 Remove label

Similarly to adding a label, we can remove labels by selecting the menu `"Labels"` -> `"Remove"` and entering either the label name or code of the label you want to remove in the pop-up window that appears.

7.4 Save label file

You can save the current Label file (`"Labels"` -> `"Save as"`) so that you can re-use it. A window will appear where you need to navigate to the location you want to save the file (starting in the Documents folder) After you select the directory, you can type the name.

7.5 Open label file

You can open a Label file you can open it by going to `"Labels"` -> `"Open"` in the menu bar. This will again open a navigation window for the user to select which file to open. If you have previously opened a label file, you can easily open it again by selecting it through `"Labels"` -> `"Open recent"`.

Troubleshooting

8.1 LabelChecker

Since this is the first release of the LabelChecker software, there is still a chance that the user might encounter some problems.

8.1.1 Installation

Mac users may face issues when installing the software.
If the software doesn't start after running the executable file,

- **Download and install .NET 8.0**
  `https://dotnet.microsoft.com/en-us/download/dotnet/8.0`
  Run the executable file. If it works, great! If it doesn't:

- **brew install mono-libgdiplus** (it takes some time)
  To install brew follow the install instructions on the brew website. Brew is installed via the terminal and should look similar to the following:

  ```
      /bin/bash -c "$(curl -fsSL https://<github link>)"
  ```

  Once brew is installed, when you run in the terminal

  ```
      $ brew-v
  ```

  should return:

  ```
      git version: <version code>.
  ```

  To install mono-libgdiplus paste the following command in your terminal and press enter:

  ```
      brew install mono-libgdiplus
  ```

  Run the executable file. If it works, great! If it doesn't:

- **Compile the software locally**
  - Clone the LabelChecker repository
  - Open the terminal and run

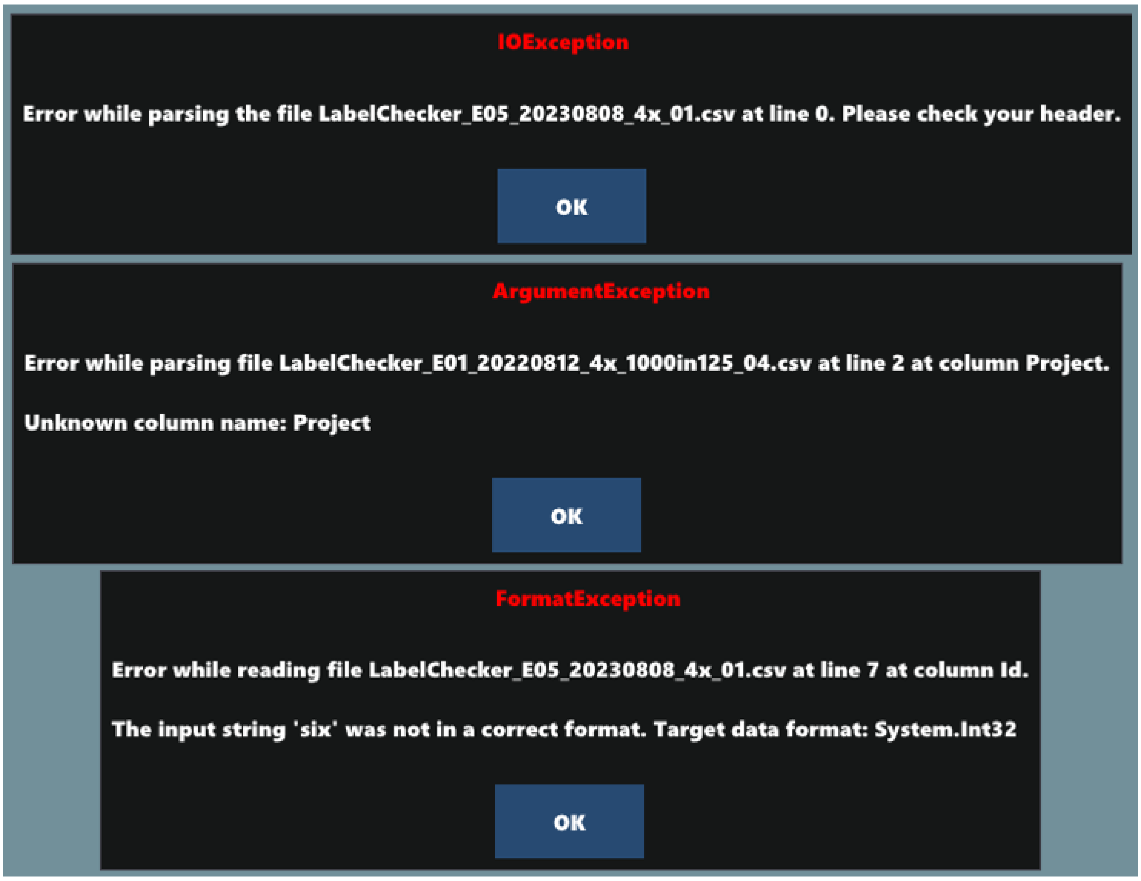
    ```
        $ chmod +x build.sh
        $ ./build.sh macos
    ```

  - Test the locally built LabelChecker Run in the terminal

    ```
        $ ./dist/macos/LabelChecker
    ```

8.1.2 Working issues

The most common issue is that the software is unable to open a LabelChecker file. When this happens, an error message is displayed to help with troubleshooting. Fig. 8.1 shows two of the most common problems; the file has a separator other than a comma (,) or a semicolon (;), and the data-type format does not match the expected one. In this case the error also returns the position in the file that caused the error.



IOException

Error while parsing the file LabelChecker_E05_20230808_4x_01.csv at line 0. Please check your header.

ArgumentException

Error while parsing file LabelChecker_E01_20220812_4x_1000in125_04.csv at line 2 at column Project.

Unknown column name: Project

FormatException

Error while reading file LabelChecker_E05_20230808_4x_01.csv at line 7 at column Id.

The input string 'six' was not in a correct format. Target data format: System.Int32

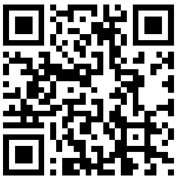
Figure 8.1: Three example error messages when opening LabelChecker files

Another possible inconvenience is when the user navigates the open file window too fast. This might cause the file browser window to close.

In the case of unexpected errors and crushes and to prevent any data and work loss, LabelChecker files are autosaved every 5 minutes. The last save is noted in the bottom right corner of the window.

8.2 Support

We provide support for the software through our Discord server. You can join by clicking here: LabelChecker or scan the QR-code:



Appendix

9.1 Keyboard shortcuts

- LabelChecker file:
  - `Open:` ctrl + O (Windows, Linux) or ⌘ + O (MacOS)
  - `Save:` ctrl + S (Windows, Linux) or ⌘ + S (MacOS)
- Image grid navigation
  - `Next page:` → or
  - `Previous page:` ←
- Object selection
  - `Multi-select:`Apply the same label to multiple objects by pressing and holding the ctrl (Windows, Linux) or ⌘ (MacOS) while clicking the images (see Fig. 6.4). The label pop-up will appear after releasing the key.
  - `Sequence select:`Select multiple adjacent objects by pressing and holding the ⇧ and selecting two objects. All objects in between the two selected objects will also get selected. The label pop-up will appear once the key is released.
  - `Select all:` Select all objects by pressing ctrl + A (Windows, Linux) or ⌘ + A (MacOS).
  - `Confirm all remaining:` A
- Undo and redo
  - `Undo:` Undo assigning of a label with ctrl + Z (Windows, Linux) or ⌘ + Z (MacOS)
  - `Redo:` Redo assigning of a label with ctrl + Y (Windows, Linux) or ⌘ + Y (MacOS)

9.2 LabelChecker file parameters

In the table below you can find all parameters in the LabelChecker file and there description. More detailed descriptions of the parameters can be found in the FlowCam user manual. datetime: data in a date and time format; float: numeric value with decimals; int: integer numeric value; string: character value.

Table 9.1: Parameters extracted from the FlowCam output, their data type and description.

Column name	Data type	Description
Name	string	Sample name
Date	datetime	Data creation date
Time	datetime	The time that the image was captured
CollageFile	string	Name of the collage file the image is stored into *Visualspreadsheet versions <6*
ImageFilename	string	Name of the image file *Visualspreadsheet versions* $\geq$ *6*
Id	int	Object id
GroupId	int	Object id in a FlowCam run/set/library/class
Uuid	string	Universal Unique Identifier; constant alphanumeric value
SrcImage	int	Original frame number where the object was captured in
SrcX	int	X coordinate where the original object image was captured
SrcY	int	Y coordinate where the original object image was captured
ImageX	int	Most left coordinate of image, either 0 or location in collage image
ImageY	int	Most top coordinate of image, either 0 or location in collage image
ImageW	int	Width of the object image
ImageH	int	Height of the object image
Timestamp	datetime	The time the image was captured in milliseconds
ElapsedTime	float	Time passed since starting the sample run
CalConst	float	Pixels to mm ratio
CalImage	int	Image used for background subtraction
AbdArea	float	Number of pixels in the threshold (binary) greyscale image converted to a measure of area by use of the calibration factor. Calculated from diameter_abd
AbdDiameter	float	Area Based Diameter: The diameter based on a circle with an area that is equal to the ABD Area(based on the binary image overlay)
AbdVolume	float	Sphere volume calculated from ABD Diameter

Table 9.1: Parameters extracted from the FlowCam output, their data type and description. (continued)

Column name	Data type	Description
AspectRatio	float	The ratio of the axes' lengths of the ellipse with a center at the particle's centroid and with the same geometrical moments up to 2nd order (Legendre ellipse of inertia) Perfect circles and squares have a value of 1.0, while near-zero values indicate long, thin particles. Reference: ISO 9276-6:2008
AverageBlue	float	Average blue pixel intensity
AverageGreen	float	Average green pixel intensity
AverageRed	float	Average red pixel intensity
BiovolumeCylinder	float	Object cylinder volume
BiovolumePSpheroid	float	Object prolate sheroid volume
BiovolumeSphere	float	Object sphere volume
Ch1Area	float	Curve area of chlorophyll 1; for FlowCams equipped with laser
Ch1Peak	float	Highest value of chlorophyll 1; for FlowCams equipped with laser
Ch1Width	float	Curve width of chlorophyll 1; for FlowCams equipped with laser
Ch2Area	float	Curve area of chlorophyll 2; for FlowCams equipped with laser
Ch2Ch1Ratio	float	Log10 (Ch2Peak) - Log10(Ch1Peak). (real $\geq$ 0); for FlowCams equipped with laser
Ch2Peak	float	Highest value of chlorophyll 2; for FlowCams equipped with laser
Ch2Width	float	Curve width of chlorophyll 2; for FlowCams equipped with laser
Ch3Area	float	Curve area of chlorophyll 3; for FlowCams equipped with laser
Ch3Peak	float	Highest value of chlorophyll 3; for FlowCams equipped with laser
Ch3Width	float	Curve width of chlorophyll 3; for FlowCams equipped with laser
CircleFit	float	how closely the object fits in a circle; values from 0-1, with 1 being a circle
Circularity	float	Parameter computed from the perimeter and the AreaFilled. Circles have value of 1; inverse of Compactness
CircularityHu	float	Alternative measure of circularity with better indication of the circular shape for very small particles or particles with edge defects. Circles have value of 1

Table 9.1: Parameters extracted from the FlowCam output, their data type and description. (continued)

Column name	Data type	Description
Compactness	float	Parameter computed from the perimeter and the AreaFilled. More convoluted shapes have greater values. Circles have value of 1; inverse of Circularity
ConvexPerimeter	float	Perimeter of the convex hull
Convexity	float	Parameter computed as the ratio of AreaFilled to the area of the convex hull of the particle. Also called Solidity. Circles have value of 1
EdgeGradient	float	Average intensity of the pixels making up the outside border of a particle after a Sobel Edge Detect convolution filter has been applied to the raw camera image. Focused images have higher EG values compared to blurry ones
Elongation	float	The inverse of Geodesic Aspect Ratio; circles or squares have value 1; elongated particles have higher values
EsdDiameter	float	Equivalent Spherical Diameter: mean value of 36 feret measurements
EsdVolume	float	Sphere volume calculated from ESD Diameter
FdDiameter	float	The diameter based on a circle with an area that is equal to the area_filled. Based on all the pixels within the particle edge trace
FeretMaxAngle	float	Angle of the largest feret measurement
FeretMinAngle	float	Angle of the smallest feret measurement
FiberCurl	float	Parameter computed from Geodesic Length and Length. Also called Curl Index. (GeodesicLength / Length) – 1
FiberStraightness	float	Parameter computed from Geodesic Length and Length
FilledArea	float	The area represented by the object edge and the pixels inside the edge. (real >0)
FilterScore	float	The statistical filter score. (*Visualspreadsheet filters*)
GeodesicAspectRatio	float	The ratio of Geodesic Thickness to Geodesic Length. Inverse of Elongation
GeodesicLength	float	Values from modeling the particle as a rectangle and computing length and thickness from: area = GeodesicLength * GeodesicThickness and perimeter = 2 * GeodesicLength + GeodesicThickness

Table 9.1: Parameters extracted from the FlowCam output, their data type and description. (continued)

Column name	Data type	Description
GeodesicThickness	float	Values from modeling the particle as a rectangle and computing length and thickness from: area = GeodesicLength · GeodesicThickness and perimeter = 2 · GeodesicLength + GeodesicThickness
Intensity	float	Average grayscale value of the pixels making up a particle (grayscale sum / particle pixels). Brighter images have higher intensity values
Length	float	Maximum value of 36 feret measurements; object length
Perimeter	float	Length of the particle edge
Ppc	int	Particles per chain. Number of particles in a chain (based on the nearest neighbor distance)
RatioBlueGreen	float	AverageBlue/AverageGreen; for coloured cameras
RatioRedBlue	float	AverageRed/AverageBlue; for coloured cameras
RatioRedGreen	float	AverageRed/AverageGreen; for coloured cameras
Roughness	float	Irregularity (roughness) of object's surface; the ratio of perimeter to convex perimeter
ScatterArea	float	Curve area of scatter laser; for FlowCams equipped with laser
ScatterPeak	float	Highest value of scatter laser; for FlowCams equipped with laser
SigmaIntensity	float	Standard deviation of greyscale values
SphereComplement	int	*deprecated sphere measurement*
SphereCount	int	*deprecated sphere measurement*
SphereUnknown	int	*deprecated sphere measurement*
SphereVolume	float	*deprecated sphere measurement*
SumIntensity	int	Sum of greyscale pixel values
Symmetry	float	Measure of the symmetry of the particle relative to its centre. Symmetric objects have a value of 1; used to identify 'broken' or partial particles
Transparency	float	Object transparency; values 0-1; 0 for a filled circle, near 1 for an elongated or irregular shape or a shape that has many interior holes. 1/(AbdDiameter/EsdDiameter)
Width	float	Minimum value of 36 feret measurements; object width
Preprocessing	string	Assigned preprocessing labels
PreprocessingTrue	string	Confirmed preprocessing label
LabelPredicted	string	Assigned classification labels
ProbabilityScore	float	Probability score for LabelPredicted
LabelTrue	string	confirmed classification labels