LUDWIG-MAXIMILIANS-UNIVERSITÄT
TECHNISCHE UNIVERSITÄT MÜNCHEN

Helmholtz Zentrum München
Institute of Computational Biology

Bachelorarbeit
in Bioinformatik

Development and Application of a Single Cell Quantification Tool for Microscopy Images

André Seitz

Aufgabensteller:	Prof. Dr. Dr. Fabian Theis
Betreuer:	Michael Schwarzfischer
Abgabedatum:	15.09.2013

Ich versichere, dass ich diese Bachelorarbeit selbständig verfasst und nur die angegebenen Quellen und Hilfsmittel verwendet habe.

15.09.2013 _____

André Seitz

Abstract

Since the development of new, high resolution microscopes enables capturing of single cells, this method provides an important approach to investigate biological properties of cells. Certain regions and molecules of a cell can be captured, using fluorophores, which are attached to the cells surface or interior molecules by antibodies. Genetic modification of certain regions of the DNA of cells of interest enables real-time image capturing in certain time intervals, that reflects expression of certain genes in those cells. Therefore, either the target protein can be fused with the fluorescence protein or the fluorescence protein gets co-expressed without linkage. The certain images of a time-lapse experiment can be concluded to a movie and provide time resolved data for further analysis. Fluorescence intensities of the certain cells can be quantified and further be regarded as relative protein amounts. Using the measured intensities, derived amounts of different proteins, and their correlations, regulatory networks may be inferred. This is applied for example in investigation of embryonic stem cells (ESCs). ESCs are pluripotent, self-renewal, can be cultured in-vitro over a long period of time, and can artificially be stimulated to undergo differentiation. By understanding the biological backgrounds of ESC pluripotency and differentiation, these cells promise to be used even better for drug development and investigation of widespread diseases like diabetes and cancer.

In scope of this work, we developed a tool for automatic single cell quantification in microscopy images (*SCQMI*), which can also be applied on time-lapse movies of fluorescence images. To be usable by everyone, it provides an intuitive graphical user interface. A lot of use cases can be covered by *SCQMI*, as the generic workflow, which is processed prior to the quantification, can be easily defined by users themselves. Furthermore, *SCQMI* is capable to handle very big datasets up to whole imaging experiment's data, spanning over multiple dimensions, like fields of view, timepoints, wavelength and focal planes. A worflow, once generated, can be automatically performed on a lot of selected images of the experiment. After development, *SCQMI* was successfully applied on two experiments, revealing new insights in the mechanisms of ESC pluripotency. Transcription factors, thought to be hubs of the pluripotency associated regulatory network were simultaneously imaged during these experiments. A workflow, including background correction, segmentation, and quantification, was created and performed on the images. The resulting large-scale datasets were analysed and compared against findings in literature. Last but not least, we present some findings, not yet described in literature, and propose certain further investigations.

Zusammenfassung

Seit die Entwicklung neuer, hoch auflösender Mikroskope die Aufnahme von einzelnen Zellen ermöglicht, bietet diese Methode eine wichtige Herangehensweise zur Untersuchung der biologischen Eigenschaften von Zellen. Durch fluoreszierende Farbstoffe, die mittels Antikörpern auf die Zelloberflächen oder sogar in die Zellen gebracht werden, können bestimmte Bereiche und Moleküle der Zellen gezielt erfasst werden. Die gentechnische Veränderung bestimmter exprimierender Bereiche der DNS von zu untersuchenden Zellen, ermöglicht eine durchgehende Bildgebung in Intervallen, die die Expression bestimmter Gene in einzelnen Zellen widerspiegelt. Dies kann sowohl durch Fusionierung des Zielproteins mit dem Fluoreszenzprotein, oder aber durch eine Koexpression des Fluoreszenzmarkers zusammen mit dem Zielprotein erfolgen. Die einzelnen Bilder des Experiments können zu Zeitraffer-Filmen zusammengefasst werden und bieten somit zeitaufgelöste Expressionsdaten zur weiteren Analyse. Die Intensitäten der Fluoreszenz einzelner Zellen können quantifiziert und im Weiteren als relative Proteinmengen betrachtet werden. Mittels der gemessenen Intensitäten verschiedener Proteine und deren Korrelationen können regulatorische Netzwerke abgeleitet werden. Dies findet unter anderem Anwendung bei der Untersuchung von Embryonalen Stammzellen (ESZs). ESZs sind pluripotent, selbsterneuernd, können über einen sehr langen Zeitraum in-vitro kultiviert werden und jederzeit künstlich zur Differenzierung angeregt werden. Durch das Verständnis der Funktionsweise von ESZs erhofft man sich, diese Zellen noch besser für die Entwicklung von Medikamenten und die Erforschung von Krankheiten wie Diabetes und Krebs verwenden zu können.

Im Rahmen dieser Arbeit haben wir *SCQMI* (single cell quantification in microscopy images), eine Software für die automatische Quantifizierung von einzelnen Zellen in Mikroskopie-Bildern, entwickelt. Das Programm bietet eine intuitive graphische Benutzeroberfläche und kann mit sehr umfangreichen Datenmengen umgehen, bis zu Bilddaten von ganzen Experimenten, die sich über viele Dimensionen, wie Sichtfelder, Zeitpunkte, Wellenlängen und Fokusebenen, erstrecken. Viele Anwendungsfälle werden von dem Programm abgedeckt, da der generische Workflow, der vor der Quantifizierung ausgeführt wird, leicht selbst durch den Nutzer erstellt werden kann. Ein einmal erstellter Workflow kann anschliessend automatisch auf vielen ausgewählten Bildern eines Experimentes angewendet werden.

Im Anschluss an die Entwicklung wurde dieses Programm erfolgreich auf zwei Experimenten angwendet und eröffnete neue Einblicke in die Mechanismen der Pluripotenz von ESZs. Dabei wurden die Proteinmengen der Transkriptionsfaktoren, die als wichtige Knoten des regulatorischen Netzwerkes angesehen werden, gleichzeitig gemessen. Dazu wurde ein Workflow, bestehend aus Hintergrundkorrektur, Segmentierung und Quantifizierung erstellt und auf den Bildern angewendet. Der daraus resultierende Datensatz, bestehend aus Messwerten der einzelnen Zellen, wurde analysiert und mit Erkenntnissen aus der Literatur verglichen. Letztlich präsentieren wir einige neue Erkenntnisse und schlagen Experimente vor, um diese Ergebnisse weiter zu untersuchen.

Contents

List of Figures

List of Tables

1 Introduction

1.1 Motivation

Embryonic stem cells (ESCs) are pluripotent and self renewal [4] and can be kept in this state even in-vitro over a longer time [33]. Therefore they are in focus of science, since they can be cultured. As these cells can undergo multi-lineage differentiation [26] there is great interest in controlling the differentiation of these ESCs, to create specifically differentiated cells for drug development and cell replacement based therapies. During the certain investigations of ESCs, unusual proliferation properties were detected. Although ESCs remain ESCs and do not undergo differentiation, they show strong gene expression heterogeneities [2, 32, 17, 6, 13]. These gene expression heterogeneities can not be investigated with population wise methods, but only with approaches, that regard single cells or specific colonies. Several approaches were developed, to investigate the gene expression behaviour of ESCs, as well as the regulation by transcription factors.

Protein regulations are often derived from gene regulatory networks, as every gene codes for a certain protein and can be regulated by other proteins [30]. The methods to reveal gene regulations and therefore possible protein regulations, like ChIP-chip approaches [62] or synthetic genetic array analyses [24] are expensive and do not provide absolutely reliable data, as the protein-gene interactions for example of the gene regulatory network has to be interpreted, eventually considering other data sources, like physical protein-protein interaction data. As the derivations are therefore not completely reliable, the fluorescence imaging approach provides a good alternative. The fluorescent imaging approach, indeed, cannot be taken into account as a high throughput method, if cells have to be imaged time lapsed. For time lapse imaging, the proteins of interest, or their coding DNA, have to be linked with a fluorescent marker. Then the cells can be cultivated, put on a medium, and the time lapse movie made, by taking an image iteratively after some period of time. The resulting data shows, in contrary to gene regulation data, real correlations between the proteins, meaning if two proteins, which are correctly marked, correlate, they are either both up- or down-regulated. This up- or down-regulation can directly be quantified, by measuring the cell's intensities in the taken pictures. Another advantage of fluorescence imaging is, that it uses single cell informations instead of population means. Therefore, correlations between certain proteins can be quantified for every cell, offering an additional information content, that can be used for further analysis and interpretation. Also, the previously named expression heterogeneities can only be observed between single cells or colonies, not between whole populations, which is why the imaging approach has to be favoured.

The non-high-throughput character of time lapse fluorescence imaging means, that the scientists have to try to detect hints for correlations between proteins in literature or by other (high-throughput) experiments first. Also, after the first analysis of fluorescence imaging experiments, it is required to decide what experiment to perform next, taking the results of the previous into account. With the tool, arising from this work, or its future versions, we provide the scientists a possibility to efficiently perform a lot of image processing steps, which will produce numeric data ready for analysis and help to decide what experiments to perform next. This will facilitate and speed up the so called system biology loop. Additionally, many more image data can be quantified, using *SCQMI*, than manually, due to the tool's automatic mode, which also produces highly reliable data (as

described in Section 3).

With the data and insights gained from single cell fluorescence imaging and other methods, one attempts to understand and explain the connection between the observed proteins and therefore the respective biological system. Investigations of pluripotency factors in embryonic stem cells, for example, hopefully reveal the underlying concepts of pluripotency and differentiation. As cancer cells are pluripotent, those findings may be used to develop new medical treatments, which specifically tackle pathological cells. Additionally the findings might help to differentiate the cells more targeted, providing a lot of specific cells for further experiments, like drug development. Regarding diabetes and cardiovascular diseases, also exemplary for widespread diseases, there is strong interest in the quantification of specific exogenous factors, that influence the course of disease, like stress, drugs or nutrition.

Of course, the developed tool has to perform all tasks, as correcting fluorescence imaging errors, detecting cells, and quantifying the selected objects, correct, to provide reliable results. In the following, all considerations and steps to achieve this, in scope of this work, are shown. In particular, we first collect already existing tools in this field and consider all required specifications in Section 1.3 and Section 1.4. Next, we present our development efforts for the post-acquisition tool in Section 2. The data-structure and the functionality for generating a workflow are explained in especial detail. The validation and evaluation of parts of the resulting tool (see Section 3) proves, that they perform as expected and supply useful data. The developed tool was then used on real experiments' fluorescence time lapse movies of embryonic stem cells (see Section 4), trying to reveal new insights in the correlations and functions of proteins, suspected to regulate self-renewal and differentiation. At last, we give a brief conclusion and outlook of the field in Section 5, regarding the use of fluorescence imaging, its associated computational methods and tools, and the investigation of embryonic stem cell proliferation.

1.2 Bio-Imaging

Since the first microscopy images were captured in 1891 [54], a lot of improvements were achieved in this field. Nowadays, time lapse microscopy movies can be automatically taken in different wavelengths and different fields of view of a sample. But still there remain some major limitations (see Figure 1, [12]).

Depending on the technique of the microscope, light emits from under or above the sample, which is fixed on a light permeable medium, passes through the sample, getting more absorbed the thicker a specific region is, and reaches a detector, like a camera chip after it got magnified by several lenses. If normal spectral light is used, this technique is called bright-field imaging. Another method is to shine light onto the sample and capture the reflections or fluorescence of the sample. Light of specific wavelengths and special fluorescence markers are often used to visualise the sample, hence this method is called fluorescence microscopy. In the case of single cell microscopic imaging, adequate magnification is always needed, as the cells only measure about 10 μm. As cameras with a resolution high enough to capture the whole medium with one image do not exist, the whole area has to be divided into multiple fields of view, called positions, and captured one after the other. This is normally done automatically by a robot, which moves the medium below the optics. To also take account of the cell's behaviour during time, so

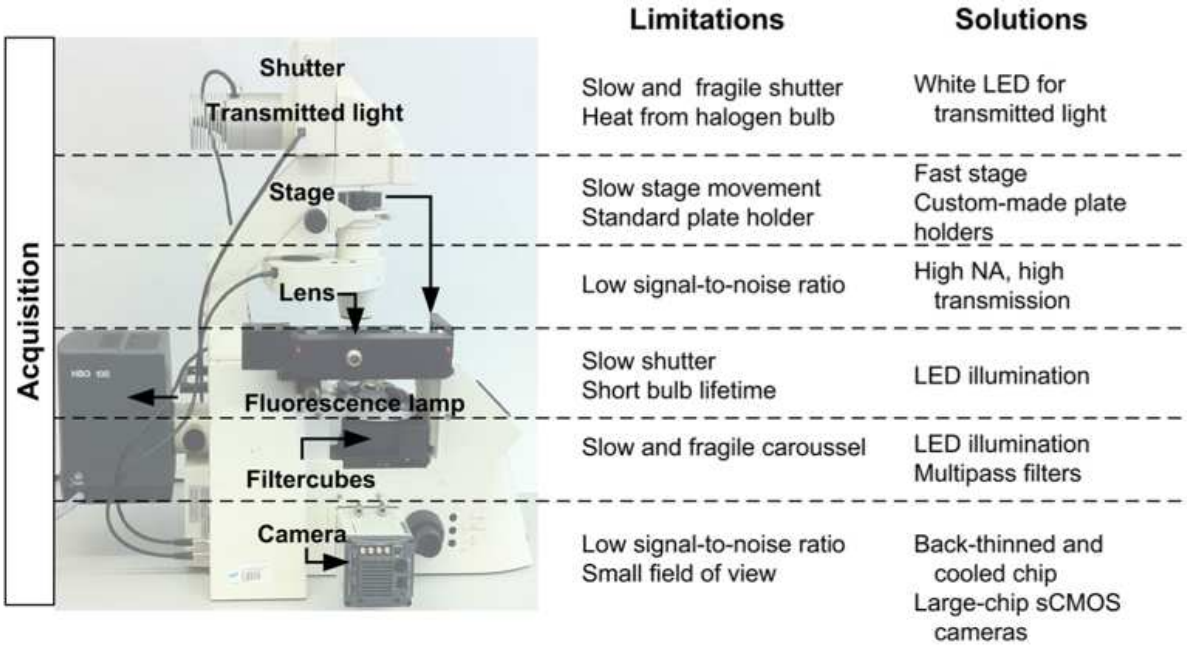

Figure 1: *For image acquisition of fluorescence or bright-field images, microscopes, like visualised here, are used. Light emits from a light-source, like a normal halogen or LED lamp, or a special fluorescence lamp. To get illumination right, a shutter regulates the time, the sample is exposed to the light. The medium, the sample is applied on, is fixed on the so called stage. For multiple fields of view, the stage can automatically be shifted between light emitter and optics in three axis. The optics part of the microscope contains some lenses, which magnify the field of view and provide certain filter capabilities. Last but not least the light gets captured by a camera (image adapted from Coutu and Schroeder [12]). All these parts have limitations, that can partly be countered. One major improvement of the last time was, for example, the use of LED backlights for microscopy. As LEDs do not produce as much heat as other light sources, the sample is not that influenced as much during the imaging. Another advantage is, that light emitted by LEDs is brighter and more even, what directly improves object detection in the post-acquisition step.*

called time lapse movies are taken from the samples, by capturing an image continuously after a certain period, which is called time point.

Altogether, it is a field on its own, just to take images even of already prepared cells [36], as there are some major drawbacks in this imaging approach. For example, exposure of living cells to light of specific wavelengths damages the cells and influences further measurements. Heat, produced by the light emitters and the robot, which moves the sample table is another problem, as it also influences cell growth and, in worst case, leads to cell death. Some cellular markers cannot be linked efficiently with a fluorescent chemical or protein [20], as the linkage means either a bias of the marker's biological behaviour, what makes measurements of living cells unusable, or -worse- cell death. So one has to visualise them by so called immuno-fluorescence staining. Therefore, the cells have to be fixed on the medium and the markers of interest are stained by antibody-fluorescence complexes. The step of fixating the cells on the medium during a time lapse movie is impossible, as the fixation is irreversible and the cells no longer viable after the procedure.

Although all these drawbacks have to be taken into account, single cell imaging is a widely used method [31, 14], as it promises direct looks on living cell data and, using fluorescence microscopy, into their molecular behaviour. As there are only few direct competitors to the classic quantitative fluorescence imaging in the area of small molecule measuring of living cells, new approaches, using fluorescence imaging, are developed permanently [22, 10, 25]. Some derivatives of the classic fluorescence imaging, like multi photon flow cytometry [57] and quantum dots [55] also achieve good imaging and quantification results, but are much more complex and expensive. As with the growing field of automatic fluorescence imaging, also the claims for fitting post-acquisition methods are expanding, we want to meet these needs with the methods and tools developed in scope of this work.

1.3 Overview of Existing Tools

As single cell imaging approaches produce huge amounts of biological data, there is an increasing demand for computational tools, which are able to process that data. Exemplary, time lapse movies, containing several wavelengths and multiple positions, allocate memory from about 50 GB up to 1 TB on hard drive. Although, one can find a myriad of publicly available image processing tools, like *Marvin* [1], *medipy* [53], the *Phyton Morphology Toolbox* [40], and many more, those are individually hardly useful, because they are mostly written to match one or few specific use cases and file formats. As the use cases, as well as the needed results and measurements, are highly variable, an integration of all functions in one tool would make the software way to complex to be developed or even used. Still, there are some approaches to collect useful functions and relevant pre- and post-processing steps, like data conversion in tools and to provide those functions to users by more or less simple graphical user interfaces. Some of the tools, which are widely used, are collected and described in the following.

Cell-Profiler **[5]** *Cell-Profiler* has a nice graphical user interface, which enables the user to simply assemble an image processing pipeline. All typical steps of image processing are predefined in *Cell-Profiler*, as well as further special functions, which are offered by a huge active community. Any image processing pipeline, once defined in *Cell-Profiler*, can be used on a large-scale dataset. Therefore, *Cell-Profiler* pro-

vides grid interfaces and associated monitoring capabilities. Functions, which are not provided in Cell-Profiler can be added by developing them in Python. C and Java code can also be integrated by usage of native libraries. Cell-Profiler and all of its additions are distributed under GNU General Public License and so available for free.

ImageJ/Fiji **[45]** **Fiji** is just ImageJ, which is expanded by some basic functions and a bunch of community provided plugins, which are validated by supervisors. *ImageJ/Fiji* itself is basically an open source image manipulation program like *GIMP*. Users can load images with the graphical user interface and perform various actions on this data. Developers can extend *ImageJ/Fiji* by implementing a plugin in Java, the software's development language. To assemble functions to a complete processing pipeline, one can use a build-in macro recorder. This recorder captures every action a user performs on the active image and lists them in an editor. *ImageJ/Fiji* also provides a headless mode, which can be used to process images with a predefined macro.

MATLAB - Image Processing Toolbox [35] MathWorks' *MATLAB* is a technical programming language and provides an Image Processing Toolbox. All important functionalities, like data handling, conversions, segmentations, and lot of others are provided. Thanks to the expensive graphical user interface for development and its capabilities, like syntax high-lighting and code completion, even rather inexperienced users can quickly create a pipeline which achieves all requirements. A pure graphical user interface to somehow assemble a workflow with some mouse clicks is not provided, but can be built by oneself with *MATLAB*'s tools. Extensions can also be developed easily, as *MATLAB* provides raw data access as well as lot of basic functionalities, which can be assembled and packed to new tools. *MATLAB* can be used on all platforms and the platform independent code can be interchanged between them easily. One big drawback is however, that the functions perform rather slow. Although students can mostly use it for free, a normal user has to buy an expensive license to use it.

OpenCV [3] The Open Source Computer Vision library was created to speed up a lot of computer vision applications and machine learning software. As it is a library and therefore supporting developers with implementations of a lot of basic functions, it does not provide a graphical user interface on its own. The library can be used in C, C++, Java and Phyton. Especially in system-near programming languages, like C and C++, its functions are high performance. It is free for both academic and commercial use, as it is released under a BSD license.

Every tool has its special purpose, it was developed for, and therefore its own advantages and disadvantages. As no tool fits completely our expectations (see Section 1.4), we decided to develop a new one on our own. Therefore, we wanted to use one of the presented tools as framework, or to integrate the functions into. In the following we present and explain the specifications, we collected first, to spend our development effort in the right direction and to keep effort minimal.

1.4 Tool-Specifications

Although there exist some tools providing important functionalities of data analysis of microscopy images and especially single cell quantifications, like background correction, segmentation, and tracking, still the single steps mostly had to be connected manually. This time consuming procedure was simplified in one single tool (*SCQMI* - single cell quantification in microscopy images). Therefore, firstly the specifications and use cases were collected and discussed in collaboration with scientist, who create, measure and process microscopy image data from biological experiments and will use *SCQMI* on daily basis.

Graphical User Interface As the future users have somehow low affinity to command line or complex programs, *SCQMI* had to provide an easy to use interface which allows to efficiently perform all desired tasks. Additionally, all functions had to be explained in tool tips in the graphical user interface, that pop up on mouse-over.

Framework Integration *SCQMI* had to be included in an existing framework, already providing basic functions of the analysis workflow, to keep programming effort in limit. Framework integration also had to guarantee the user the possibility to develop his own workflow by integrating own functions into a fix frame of image analysis, containing background calculation, correction, segmentation and quantification. These additional functions had to be provided by the framework, as otherwise the developing effort would have exceeded the scope of this work.

Platform Independency To ensure access as easy as possible to every potential user, *SCQMI* was thought to be platform independent or at least usable on all major platforms. Therefore it had to be developed in a platform independent language and all included tools had to be usable on all platforms, too.

Workflow Creation and Application The functionalities, the framework provides, and those we added, have to be assembled to a workflow, which can simply be performed on one or more images. All type conversions, in- and output and other compatibility issues have to be handled by *SCQMI*. The workflow has to run either automatically over all selected images of a dataset, or semi-automatically, meaning, that the workflow requires input, for example parameters, or manual refinishing for some crucial steps.

Scalability *SCQMI* needed to be highly scalable, as it had to be able to process microscopy image data from single images to whole experiment image stacks, containing several positions, time points, focuses, and wavelengths. On the one hand *SCQMI* had to be scalable in the mean of data storage, as the experiments, it had to process are very big and therefore had to be managed appropriate. On the other hand, the workflows have to be processed in an acceptable time, which means, that *SCQMI* had to provide a workflow execution in parallel on different data.

Open Source To ensure preferably wide acceptance, the framework and all used parts had to be licensed under the principles of open source and further distributed as such themselves.

Regarding these specifications, *OpenCV* obviously lacks any user interface or even a serious possibility to create one, as a developer has either to use the Qt framework or create all components on his own. Another disadvantage of *OpenCV* is, that it was

thought to be used as a C or C++ library, indeed provides interfaces for some other programming languages, as Java, but needs a platform specific compilation to be used properly.

MATLAB also does not provide a graphical user interface for users, but an very sophisticated one for developers. Additionally it is not really fast, as it was developed as a technical high level programming language. Another disadvantage is, that *MATLAB* indeed provides a lot of functionalities, but, due to the lacking frontend, no possibility to simply generate a pipeline with them.

Cell-Profiler, indeed, meets nearly all specifications, but has one major drawback, suspending itself from the list of possibilities: a workflow, once generated can not be changed or adapted for single images. As the single cell imaging data is not only very large, but also very diverse between different wavelengths, positions, or time points, we had to meet this diversity by providing the user a semi-automatic mode for bulk execution. As *Cell-Profiler* does not even offer a possibility to implement this feature, we decided to use *ImageJ/Fiji* as framework.

As *ImageJ/Fiji* was developed in Java, it is platform independent and all parts of this framework are licensed under the principles of open source. It also comes with a graphical user interface, providing easy-to-use access to most of its functions. *ImageJ/Fiji* indeed is scalable up to five dimensions and also capable to efficiently process these data, but as five dimensions are not enough to cover all experiment's data, we decided to design an data-structure on our own(see Section 2.2). The image processing framework already comes with a possibility to create workflows and run them on images, but as this feature is not intuitive enough to all users, we had to redesign it too (see Section 2.4).

As a tool for single cell quantifications was already prototyped in *MATLAB* (called *sQTFy*) by scientists of the work group for Quantitative Single Cell Dynamics (QSCD) at the Institute for Computational Biology (ICB) of the Helmholtz Zentrum München, the main functionalities of *SCQMI*, were indicated by *sQTFy*. Especially the rough rack of processes, like background calculation, correction, segmentation and quantification, was adopted (see Section 2 for details). Also, the idea of an automatic or semi-automatic bulk execution on several images was adopted. Last but not least we were able to avoid problems, which occurred during development and application of *sQTFy*. For example avoiding specific bugs, memory leaks and complicated parts of the graphical user interface helped making *SCQMI* more stable and user friendly.

2 Development

To match all of the specifications, defined in Section 1.4, we decided to integrate *SCQMI* (single cell quantification in microscopy images) into the *ImageJ/Fiji* framework. As shown in Section 2.1, *ImageJ/Fiji* provides a lot of basic functions as well as a basic graphical user interface and additionally, some more sophisticated functions, as community based plugins. The standard graphical user interface is shown in Figure 2. As *ImageJ/Fiji* is just capable to work on image data, spanning maximum five dimensions, including X- and Y-axis, we developed a data-structure on our own (see Section 2.2). To assemble an image processing pipeline, a workflow recorder was developed (see Section 2.4). As background calculation, correction, segmentation, and finally quantification are performed most commonly in single cell quantification approaches, these steps were fix implemented as a rack (see Figure 7), which can be extended by user defined processes during runtime. As some of those steps were not yet included in *ImageJ/Fiji*, they had to be implemented as *ImageJ/Fiji* plugins first. The single cell quantification tool was developed, as postulated by the extreme programming methodology, which enables stepwise feedback by potential users. Extreme programming defines an iterative development process with alternating implementation and review processes. Thereby, errors can be corrected early and specifications updated as needed. The major developed program parts, as well as the underlying concepts are shown in the following sections.

2.1 ImageJ/Fiji

As *ImageJ/Fiji* provides a lot of useful functionalities in regard of image processing for single cell quantification, we decided to use it as framework. A further advantage of *ImageJ/Fiji* is, that it is widely used by scientific as well as non-scientific users. This means, that *ImageJ/Fiji* has a broad community, which is able to provide plugins, updates, and help in case of problems.

One of *ImageJ/Fiji*'s basic functionalities is the data import and export as well as data management during runtime. Besides single images, *ImageJ/Fiji* can handle so called image stacks, which are for example time series, up to five dimensions including X-, Y- and Z-axis. During import, *SCQMI* analyses what kind of pixel format to use (RGB-color, 8-bit, 16-bit or 32-bit) and converts easily between these types. Further, *ImageJ/Fiji* provides picture wide inversions, basic maths, and simple drawings on the image. Also a Gaussian filter, mean and median filter, and other user defined color filters are already included. As the image data is always displayed after every change, all needed display functionalities are implemented. An image can be magnified, displayed with overlays, and brightness and contrast enhanced, by restricting the displayed intensities. Some basic segmentations and associated functions are also predefined, for example a manual

Figure 2: *The main graphical user interface of ImageJ/Fiji already provides some basic functions, which can be applied on a currently active image.*

thresholding, automatic thresholding algorithms, watershedding, and Voronoi tessellation. A very important tool is the selection manager and its associated particle analyser (see Figure 11, ***(C)*** and ***(D)***). The first collects selections in a list and features preview and editing of these selections in the currently active image. The second allows to preset a bunch of measurements, like size, position, intensity, and many more and to print the measured selections as a list. Last but not least, *ImageJ/Fiji* comes with a lot of packaged plugins, mostly providing more sophisticated approaches for the tasks named above.

To extend *ImageJ/Fiji*, one has to use its plugin structure. Therefore an interface, named `PlugIn`, and the extending classes `PlugInFrame`, `PlugInTool`, and `PlugInFilter` are provided. Every class contains some special characteristics, which are handed down to further extending classes. `PlugInFrame` also extends AWT's `Window` and `EventListener` classes, providing *ImageJ/Fiji* window behaviour, controlled by *ImageJ/Fiji* itself. By implementing a `PlugInTool`, a new graphically applicable tool can be defined, like a selection tool or a pencil, whereas `PlugInFilter` changes image data of the whole image or the active selection, and is for example used for the calculation of a mean value in a specific region. For the development of *SCQMI*, we decided to implement the `PlugIn` interface directly, as we needed a more sophisticated controller class, that did not directly change an active image. This top level class named `S_QtFy`, can be called directly, as a standalone program, or indirectly from an already open *ImageJ/Fiji* instance via the *Plugin* → *Process* menu. `S_QtFy` handles the major parts of the software, which are data import, data display and selection, and workflow generation.

To provide an useful graphical user interface, MigLayout [18] was used as layout manager, to handle the swing components. MigLayout uses properties of the GridbagLayout manager, extending this by elements, like row and column sizes and growth factors. The open source layout manager is very easy to use and helps building complex graphical user interfaces.

2.2 Data-Structure

As the existing possibilities to handle data in *ImageJ/Fiji* were not enough to handle the huge amounts of data produced in microscopy imaging for single cell quantification, we had to develop a data-structure and the associated functions ourselves. The specifications of *SCQMI* and its data-structure were explained in Section 1.4. To meet these specifications, we developed a multi dimensional cube or hypercube, which can be dynamically initiated in respect to the number of dimensions and their size. Therefore, a tree structure, based on recursive elements, as well as a top level operator node were developed. As loading all images of a standard experiment to the random access memory would cause every normal workstation to crash, the data-structure only holds the images virtually, as their references to the location on the hard drive. To speed up image access, the data-structure additionally implements a dynamic cache, always truncating the least recently used images. The is dynamically sized in respect to the random access memory, currently available at runtime. To instantiate this data-structure efficiently, with respect to its dynamic properties, we also developed a factory class, respecting the factory method design pattern. This factory class provides easy and efficient creation of the data-structure in various scenarios, for example single file import, folder- and subfolder import and *TTT* experiment import (see Section 2.3). Timm's Tracking Tool [41, 15, 37] is a single cell

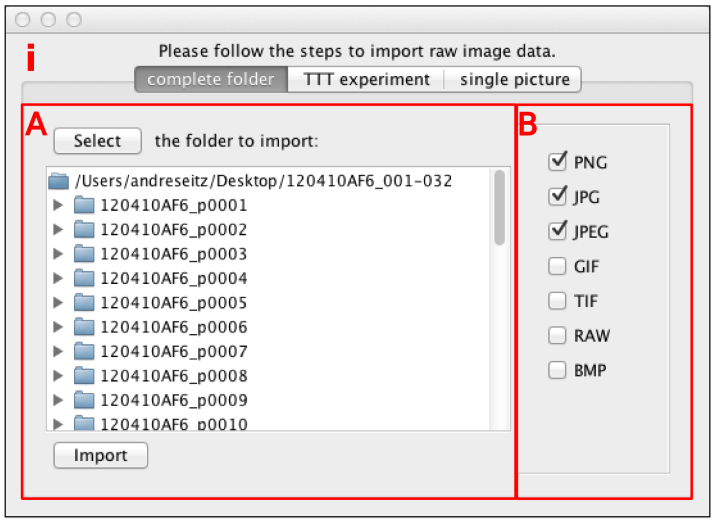

Figure 3: *(i) The bulk image import dialogue provides the ability to select a folder (the so called root path) and to preview all files and files in subfolders* ***(A)****, which match one of a few selected specific file extensions* ***(B)****. After previewing and curating the files, they can be imported into the data-structure, using the level of subfolders as dimensions.*

tracking tool, widely used by scientists, for example of the Research Unit Stem Cell Research of the Helmholtz Center Munich (SCD) and the Department of Biosystems Science and Engineering of the ETH Zurich in Basel (D-BSSE).

2.3 Interface

During instantiation of `S_QtFy`, directly after calling it, a value object, which implements the singleton design pattern, called `PassParameters` gets constructed. The singleton design pattern instructs, that the implementing class creates and holds a static instance of itself, once it gets instantiated the first time. During all following calls, the class first checks, whether it already had been instantiated and returns the existing instance. This value object holds all important parameters, that have to be passed through all other classes, for example the data-structure, user defined options and values, window sizes and further frontend related values.

After initialisation of `S_QtFy`, a data import dialogue, simply called `Import`, opens up, requesting the user to select all images that have to be processed. Therefore the user has 3 possibilities: unsorted bulk import from folders ***(i)***, import of *TTT* experiments ***(ii)***, and import of single images ***(iii)***.

Choosing the latter, one can select single files from hard drive, order them manually in any way, and import them (Figure 5, ***(F)***). The `Import` class will additionally calculate the hierarchical lowest folder, which every selected path has in common. This path, further called root path, is needed to configure references to further data locations, as well as results later.

If a user selects a root path directly after choosing the folder bulk import (Figure 3,

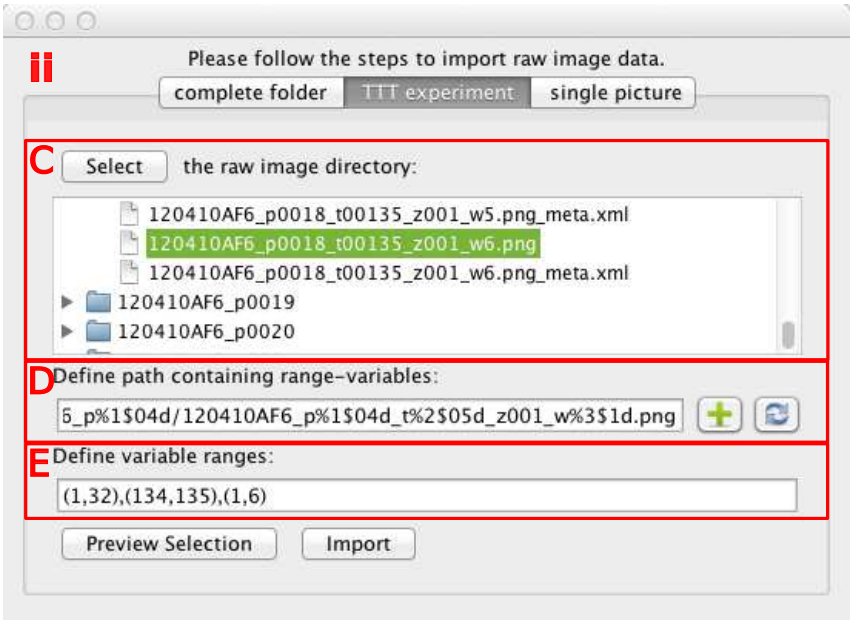

Figure 4: ***(ii)*** *The import of TTT experiments allows definition of a root path* ***(C)****, containing the raw images, ordered as defined by TTT's specifications. It provides a data preview window* ***(C)****, allowing to quickly scan the files, that will be imported. An exemplary path has to be defined* ***(D)****, containing some formatting expressions, which can be parsed by Java's* `java.lang.String` *class. The formatting expressions will be substituted by members of the numerical ranges, defined below the path* ***(E)****. On clicking "Preview Selection", the preview window* ***(C)*** *reloads, showing all files which are captured by the given path and ranges. After curation, the files can be imported.*

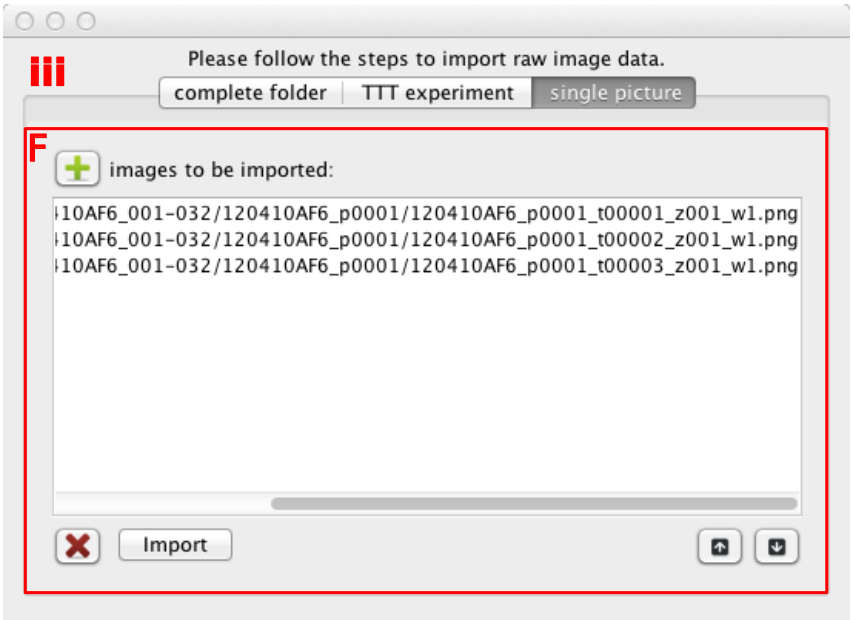

Figure 5: ***(iii)*** *Using the single image import dialogue, one can namely import a list of single images, which are selected from hard drive, into one dimension of the data-structure. Before importing the images, they can be curated and reordered in the preview window* ***(F)****.*

(A)), he also has to select some file formats (Figure 3, ***(B)***) and `Import` will screen all subfolders for files of these formats, order them in a hierarchical depth first manner, and import them.

The last import option is the import of *TTT* experiments ***(ii)***. *TTT* defines a specific folder and file name structure, which can be parsed by `Import`. *TTT*'s hierarchical data-structure was used, as the specifications of *SCQMI* were elaborated together with scientists from the ICB and D-BSSE. To parse this special folder and file name structure, one has to select a root path for raw images and to define an exemplary path to one image (Figure 4, ***(C)*** and ***(D)***). To differ between several dimensions, representing for example focuses, positions, time points and wavelengths, the path has to be manipulated in a specific manner: the dimension identifiers have to be inserted into the path, using `java.lang.String`'s formatting instructions, instead of the respective numbers of the previously selected sample file. Numeric ranges, which define the number of imported files, matching the replacement instructions above, have to be defined, too (Figure 4, ***(E)***). On demand, `Import` calculates all, to-be imported, files and displays their references in the preview window (Figure 4, ***(C)***). During the import transaction, all references are checked for their validity and stored in an adequate data-structure (see Section 2.2).

Additionally, a standard dimension is added to the data-structure during the import transaction of all import methods. This standard dimension represents and holds the data of the five main steps of a single cell quantification pipeline: raw images, background calculations, corrected images, segmentations, and the quantification results.

After importing all images, one wants to work with, `S_QtFy` switches to a data visualisation window, called `VirtualData`, as depicted in Figure 6. Firstly, one can change the standard locations (Figure 6, ***(A)***) of background calculation, correction, segmentation, and quantification results. These are predefined by *TTT*'s standard file location properties. As one might want to use previous results (for example corrections or segmentations), which are associated with the raw data, these data have to be referenced first. Therefore, immediately after instantiation of `VirtualData`, some threads (`FillWorker`) are started via a thread controller (`FillMonitor`) in the process' background, crawling for already existing data at the predefined locations and inserting them into the data-structure. `FillMonitor` and `FillWorker` were created, meeting the master-worker or master-slave design pattern. This design pattern describes the interfaces and events between the master and worker classes. On change of one or more locations in `VirtualData` (see Figure 6, ***(A)***), the `FillMonitor` organises all further steps, resulting in a new cycle of the crawling process, which leads to very fast data completion due to multi-threading.

Next, one can edit the dimension's names of the data in the data-structure (Figure 6, ***(B)***), to increase the comprehensibility in further dialogues. As one might want to select some images, which are not in the same dimension, one can reorder the dimensions and hence overlook and select data much easier. All changes are displayed in the data preview window (Figure 6, ***(C)***).

Further, one can choose the next step in `VirtualData`, which normally is the generation of a workflow. This can be performed by loading an existing one from hard drive or generating a new one after selection of an initial image (Figure 6, ***(D)***). The possibility of editing a previously loaded or created workflow is also integrated. `Workflow` is explained in Section 2.4 in further detail.

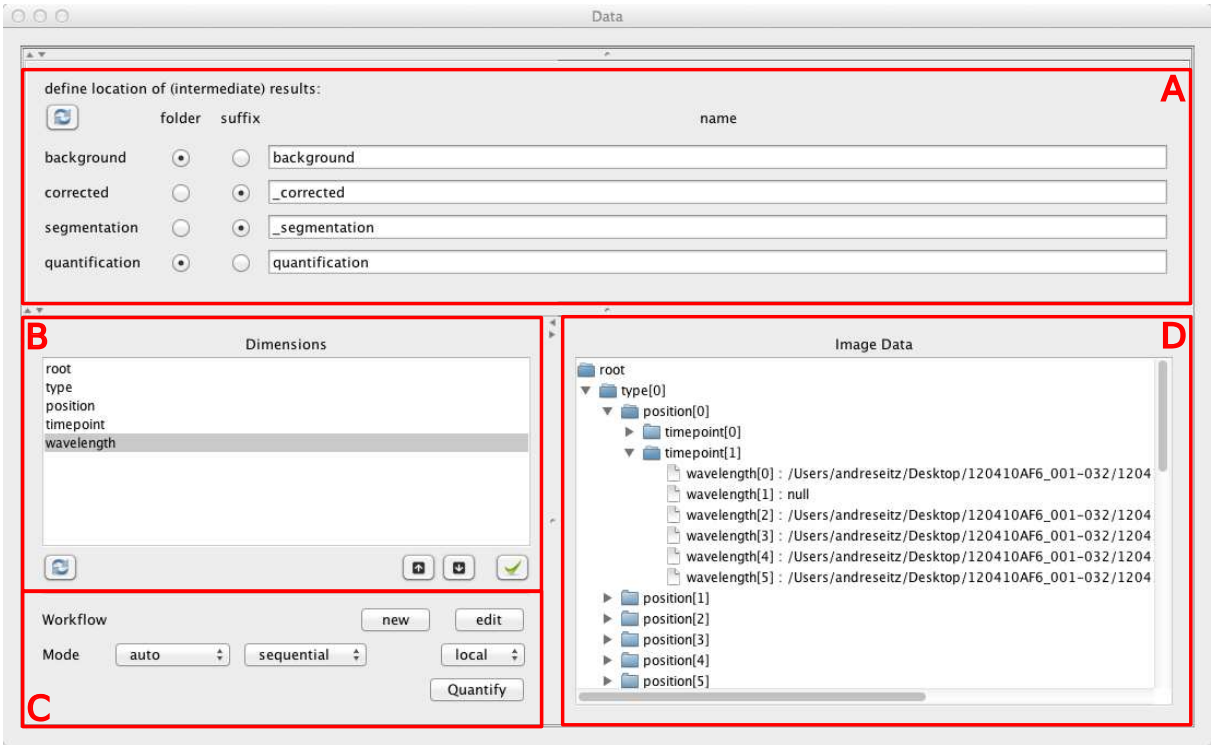

Figure 6: *The* `Virtual Data` *dialogue enables the user to define the locations of images* ***(A)****, relative to the already imported raw images, like background, corrected and segmentation images. Also the location of the quantification results have to be set here. TTT defines specific locations of background images, correction images and segmentations, which are set as default. To overview and select the data, or parts of it, as easy as possible, the dimensions can be renamed (by double clicking) and reordered as needed* ***(B)****. All changes are displayed in the data window* ***(C)****. The images, which can be used in addition to the raw images* ***(A)****, are referenced in background by several threads and added to the data-structure and therefore the data window automatically. Further steps, as workflow generation and image processing with a given workflow* ***(D)*** *can only be performed, after one or more images were selected in the data window.*

After a workflow was created, one can select some images from the image preview window and run the workflow on them. As the execution of a workflow might be computationally expensive, one was thought to be enabled to set whether the task has to run in automatic or semi-automatic mode, whether it will run sequentially or parallel, and whether the local machine or a grid will be used. In the first version of *SCQMI*, the buttons (Figure 6, ***(D)***) are disabled and a workflow can only be processed locally, in automatic mode, and sequential. The runner class, named `MacroProgress`, is further explained in Section 2.5. Both actions (workflow generation and workflow processing) require one or more selected images in the data window (Figure 6, ***(C)***).

2.4 Worflow Assembly

The workflow assembly window and the underlying operating structure is the biggest part of *SCQMI*, as its task is the most sophisticated. We assumed a standard pipeline (see Figure 7, ***(A)***) for all single cell quantification tasks, consisting of four actions: ***(i)*** background calculation, ***(ii)*** background correction, ***(iii)*** segmentation and ***(iv)*** quantification. These steps and their incoming and outgoing references were predefined during development. The images, that have to be used for a particular action, can be manually redefined during workflow generation. References to other pictures ***(B, iii)*** are also possible. On startup of `Workflow` the selected image from `VirtualData` is taken as raw image (see Figure 7, ***(A, raw)***) and displayed in a separate window, the so called preview window.

`Workflow` displays a controlling window, containing one tab for every crucial part of the single cell quantification: background calculation (Figure 8), correction (Figure 9), segmentation (Figure 10) and quantification (Figure 11). Every tab includes two drop down selections (see Figure 8, ***(B)*** and ***(C)***), which provide the predefined standard operations for the particular main part on the one and the save operation of (intermediate) results on the other hand. The most important standard operations of all four workflow steps are explained in the following section in detail. Associated to the drop down boxes are two panels, which are dynamically rendered, after the selection of the standard operation changed (see Figure 9, ***(D)*** for save parameters and Figure 10, ***(A)*** for action parameters respectively). These panels provide input elements, as some standard operations require input parameters. The standard operation is displayed as a bar in another panel, the so called pipeline list (Figure 10, ***(B)***).

By default, we set specific actions with specific parameters as standard operations, that performed very well on fluorescence imaging data, we presented in Section 4. The background is calculated by *Rolling Ball* and used for correction by the *Divide Minus One* method. The corrected image is segmented by *MSER* and quantified afterwards with the resulting segmentation mask by default. All intermediate results are saved, as long as the workflow save operations are not changed by the user.

The workflow generator also records all changes of the current active image in the preview window and displays these changes in the pipeline list of the associated tab. As *ImageJ/Fiji* does not provide this feature to developers by an interface normally, we had to get this information by catching some events fired by *ImageJ/Fiji*'s frontend system. Since not all mistakenly recorded events can be filtered, the recorder has to be used with attention. Nevertheless, all important actions can be recorded, just very special processes

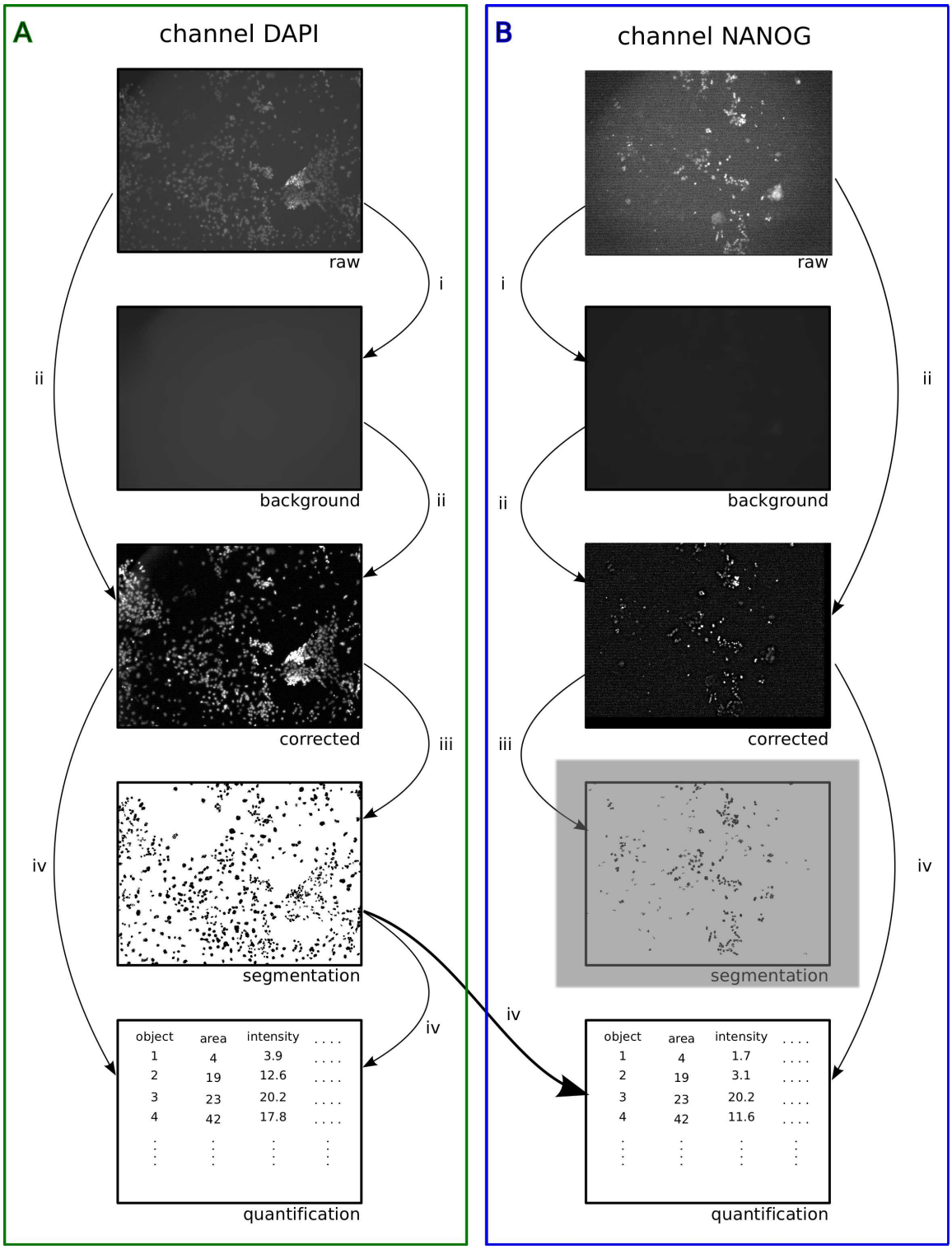

Figure 7: *A standard pipeline was assumed for single cell quantifications* ***(A)****. After calculating a background from the raw image* ***(i)****, the raw image is corrected, using this background image* ***(ii)****. The resulting corrected image is used for segmentation* ***(iii)*** *and quantification* ***(iv)****, additionally requiring a previously calculated segmentation. Though, the references for incoming and outgoing images are predefined, they can be changed during workflow generation. Also references to other pipelines* ***(B, iv)*** *are possible, covering some common use cases, for example quantifications of certain images with a segmentation from another image. Workflows, similar to this, can easily be created, using the developed graphical user interface, as shown in Section 2.4 and Figure 8, Figure 9, Figure 10, and Figure 11.*

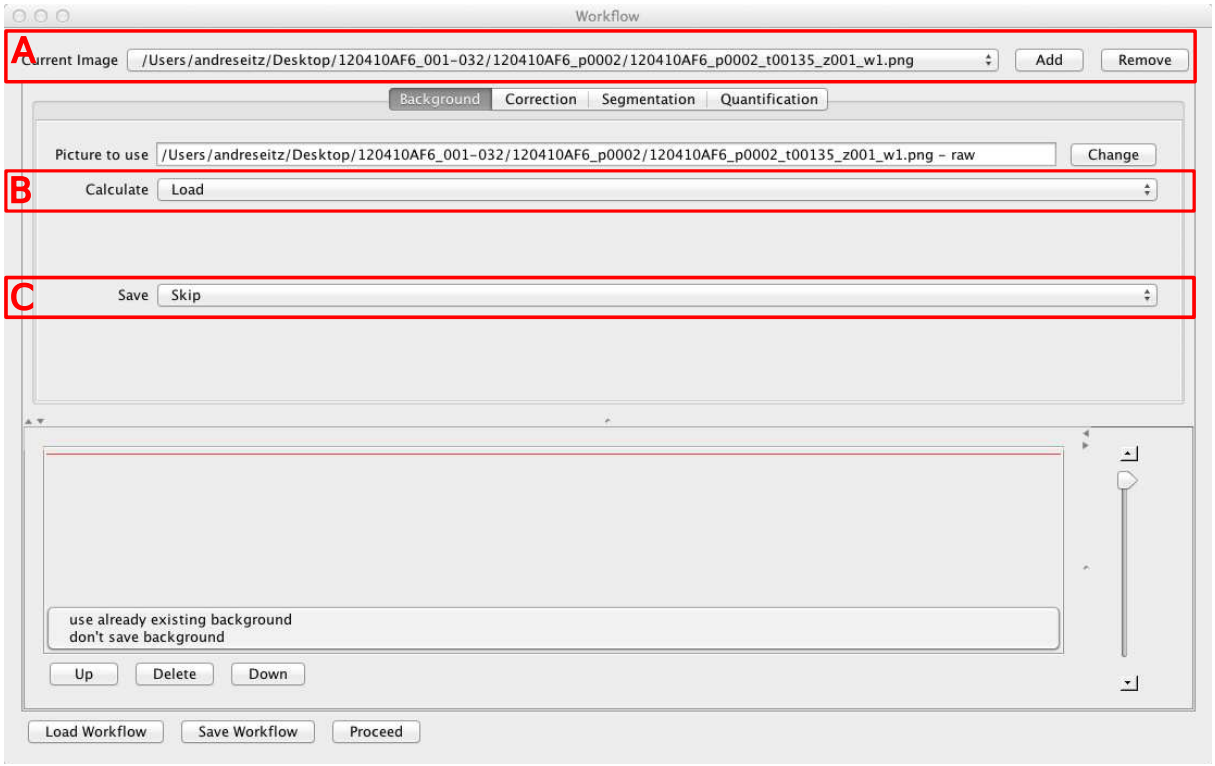

Figure 8: *The background calculation is the first step of correction and can be performed in many different ways (see Section 2.4.1).* ***(A)*** *shows the current selected picture. One can also add further pictures to the pipeline, to interchange intermediate results between the different picture's pipelines (see Figure 7). These can be selected in* ***(B)****. As after all actions, one can save the intermediate result* ***(C)****.*

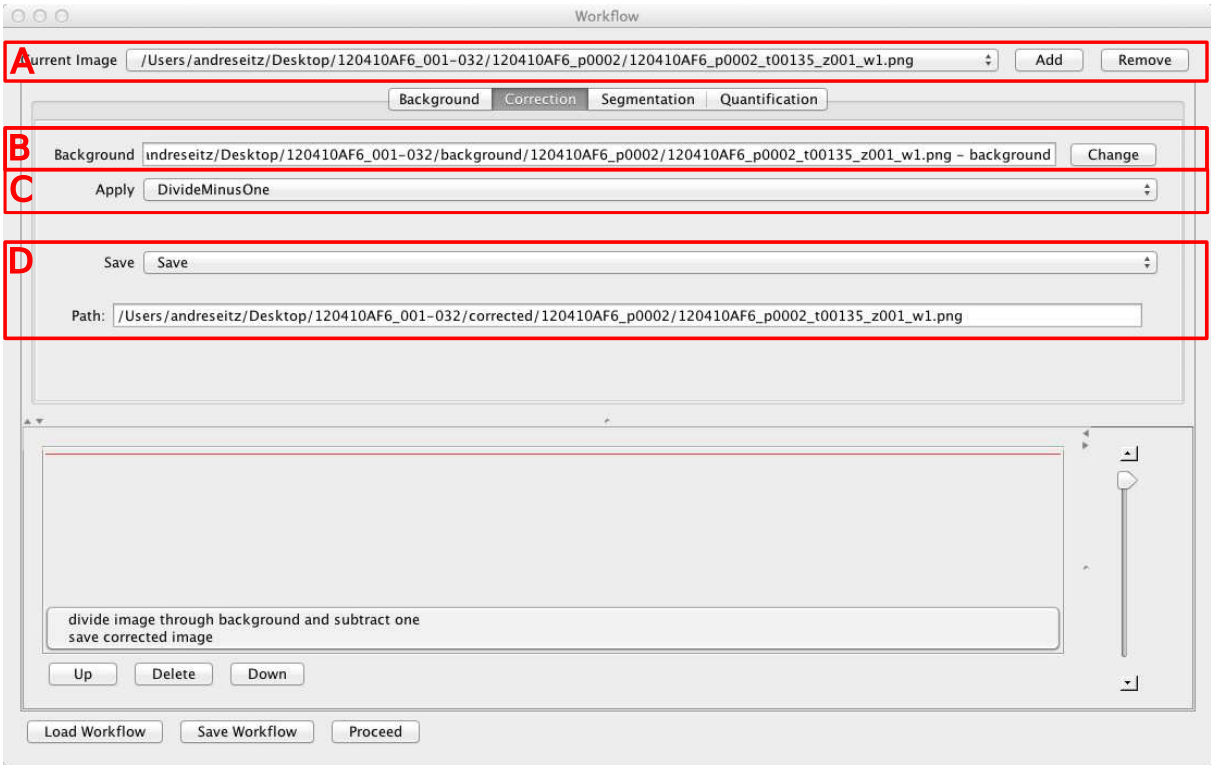

Figure 9: *(**A**) shows the current selected picture and the provided possibility to add another picture to the pipeline. Figure 7 (**B, iii**) shows one use case. The raw picture can be corrected using a background image, which can be selected in (**B**). The correction mathematically uses the background image to enhance the raw image. The kind of mathematically operation can be selected in (**C**) and is described in detail in Section 2.4.2. The result of the correction can be saved or just used as intermediate result (**D**). The panel below the selection box, shows the save location, in case the save operation was selected.*

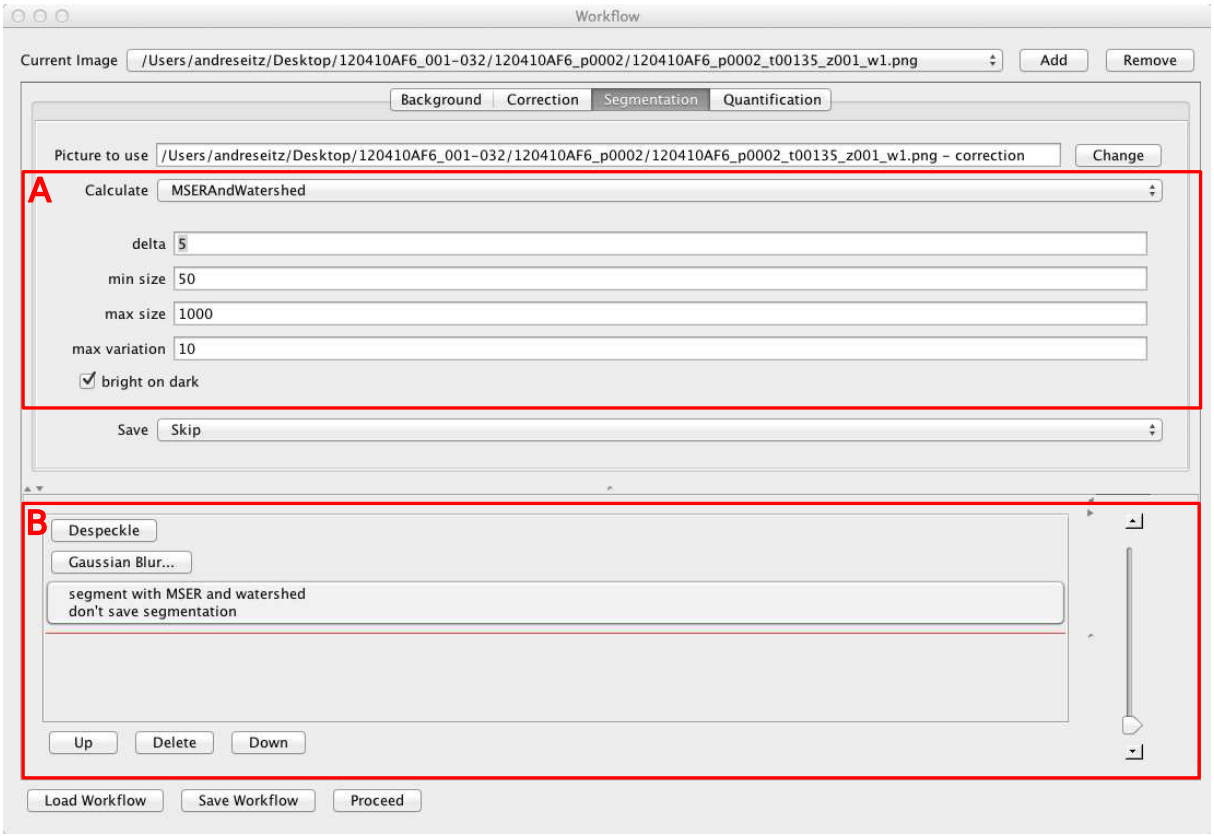

Figure 10: *Segmentation is one crucial step of the pipeline, as it is hard to achieve a valuable segmentation, depending on the quality of the corrected image. Most plugins, that are used, require parameters. Therefore a panel renders below the action selection box* ***(A)****, providing input elements for all parameters. Additional actions can be performed on the image currently previewed in the preview window. All actions are recorded and presented in a so called pipeline list* ***(B)****, providing reordering and deletion.*

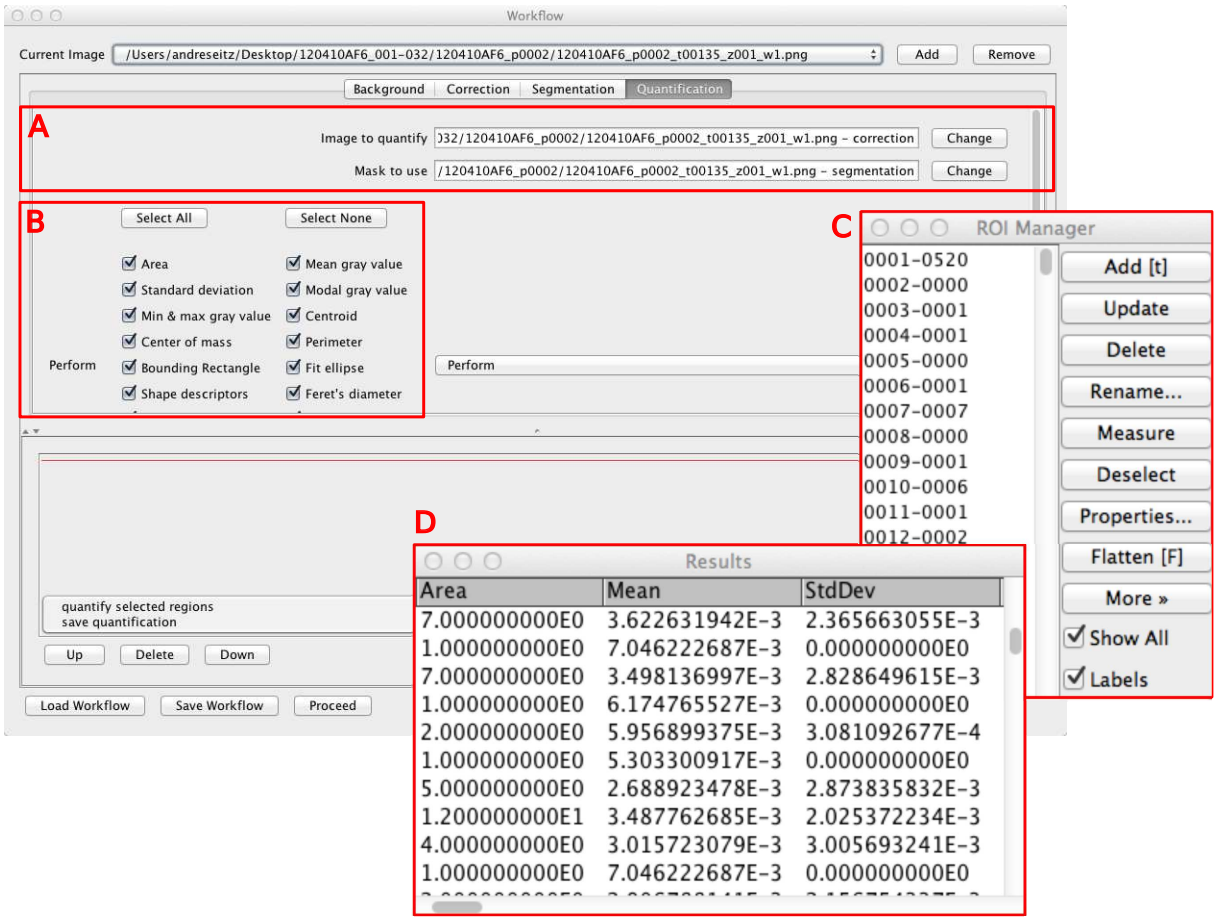

Figure 11: *Quantification and correction are actions, that require two input images (see Figure 7). For the quantification, one can set both input files* ***(A)****. The various measurements* ***(B)*** *of the quantification can be selected or deselected individually or all at once. The objects, resulting from the segmentation, chosen in* ***(A)****, are displayed in a* `ROI Manager` ***(C)****. The* `ROI Manager` *provides preview in the preview window, deletion and editing. All measured values are previewed in a* `Result` *window* ***(D)***

.

may cause an error. One can scroll up and down in the pipeline list of workflow steps, always getting the particular preview displayed in the preview window. The single steps can be rearranged and deleted, too, by simply selecting a step and clicking the specific button (Figure 10, ***(B)***). As it is required, that the save operation is always the last step of a particular workflow part, to ensure correctness, no actions are recorded, while displaying the result of the standard operation of a pipeline part. On every change, the downstream part of the pipeline is recalculated, to ensure the preview in real-time. This recalculation is firstly done directly on instantiation of `Workflow`. This is, why one has to wait some seconds, until the first interactions with the graphical user interface are enabled. The quantification step always opens some additional windows, called `Selection Manager` and `Results` (see Figure 11, ***(C)*** and ***(D)***), which are also displayed directly after startup.

As some parts of *SCQMI* use very specific image data formats, which diverge from the standard 8-bit representation, for example 32-bit values normalised between zero and one, we decided to develop an additional plugin called `ScaleConvert`. This plugin extends `ImageJ/Fiji`'s `Convert` tool. The standard conversion simply changes the value's type without scaling them. Values, which are too big or small, are simply truncated and set to the new scale's minimum or maximum respectively. `ScaleConvert` performs scaling to the new type, including the possibility, to convert not only to 32-bit with 32-bit scale, but also scale to a range between zero and one. We also decided to use this type, 32-bit normalised between zero and one, as standard type for the complete workflow, as it also enables negative values. These negative values are required, for example in background corrections, which would not work properly, if negative values would be truncated. All images, which are loaded from hard drive or calculated from other images, are converted using `ScaleConvert`. In the other direction, all images which are saved to hard drive, have to be converted to 8-bit grayscale images with a range from zero to 255. *ImageJ/Fiji* just saves the values from zero to one in a 8-bit scale, meaning a completely black image. To display the images properly, the range has to be considered here, too. Therefore, the contrast is automatically enhanced by *Process* → *Enhance Contrast ...*, every time the preview window gets actualised.

The correction and quantification steps need two input images instead of one and are handled appropriate. The background image for the correction step and the segmentation image for the quantification step respectively can be selected specifically. On top of the window one can add further pictures to the pipeline, which steps can also be selected as input for another pictures correction or quantification. One use case is, as exemplary depicted in Figure 7, the segmentation of an image, visualising a fluorescent signal from a nuclear marker ***(A)***. One might want to quantify other pictures ***(B)*** with this segmentation, too, as the segmentation, which is calculated on the second image ***(B, iii)***, is much less reliable.

All images and references of a workflow are always hold as relative references to the initially selected picture in `VirtualData`. This enables fast and valid reconstruction on reload or bulk execution. The whole pipeline can be saved, or another one loaded by clicking the specific buttons at the bottom of the window. One can also proceed to bulk execution in the same manner.

2.4.1 Background Calculation

As depicted in Figure 12, the background of microscope imaging is never equally bright (in case of bright field imaging) or dark (in case of fluorescent imaging), due to non complete permeable medium, light diffusion and other disturbing factors. One major drawback of an background biased image is, that measured intensities of objects are higher then the real intensities. As the background differs from image to image, it gets much more complicated, or even impossible, to correctly compare the results of different images. Another drawback is, that the background of one image is not equal at every position in the image. This is influenced by several factors. The light source is normally centered under the optics, so the regions at the border are differently exposed than regions in the center. This results in an image, that is brighter at the center, than towards the edges. This effect is called vignetting. The background can be mathematically estimated and corrected, to get images, which can be further processed and analysed easier. Figure 12, for example, shows an uneven illuminated raw image, with a too bright background. After correction, the foreground objects are much easier to detect, as well as by eye, as by computational methods. The intensities of the corrected image are much better distributed and foreground objects can be separated from background with a threshold, as the segmentation shows. On the contrary, the raw image's intensities are not as properly distributed as in the corrected case and a lot of cells are missing or can not be completely segmented without including too much background pixels. Some of the existing methods for background estimation were included in the single cell quantification tool as standard operations and are explained in the following:

Mean To calculate a mean background, one has to set a radius in pixel as input parameter. *SCQMI*, then, iterates over the image pixel by pixel, always calculating the mean over all pixels within the radius previously given, and sets the calculated value as new center pixel value. As dark or bright foreground objects can not be excluded, this method only results in valid backgrounds, if the majority of pixels in the image are background pixels. If foreground objects cover too much area or are clumped (see Figure 18, ***(A)***), they influence the background pixels too much leading to too high background values, again in turn, leading to biased foreground object intensities.

Median The median background is calculated the same way as the mean background, just using the median over all pixels in the given radius as center pixel value. The properties remain nearly the same. The median is said to be more robust, than the mean.

Rolling Ball [**52**] The background is calculated by virtually rolling a ball over the surface of a landscape represented by the image pixel intensities (see Figure 13). As this ball cannot slide into small but high peaks, these peaks do not influence the background as strong as they would in the mean background calculation. The radius of the ball is the main input parameter and is required to be as big as the diameter of the biggest object in the image. Flat background areas are captured very well by the rolling ball algorithm. By checking the sliding paraboloid option, one can choose to slide a paraboloid of rotation over the bottom side of the images surface. This paraboloid has the same curvature properties at its apex, as the rolling ball. The result is just approximated by sliding in four directions as a compromise between accuracy and speed. Normally, the result is smoothed by calculating the mean over a

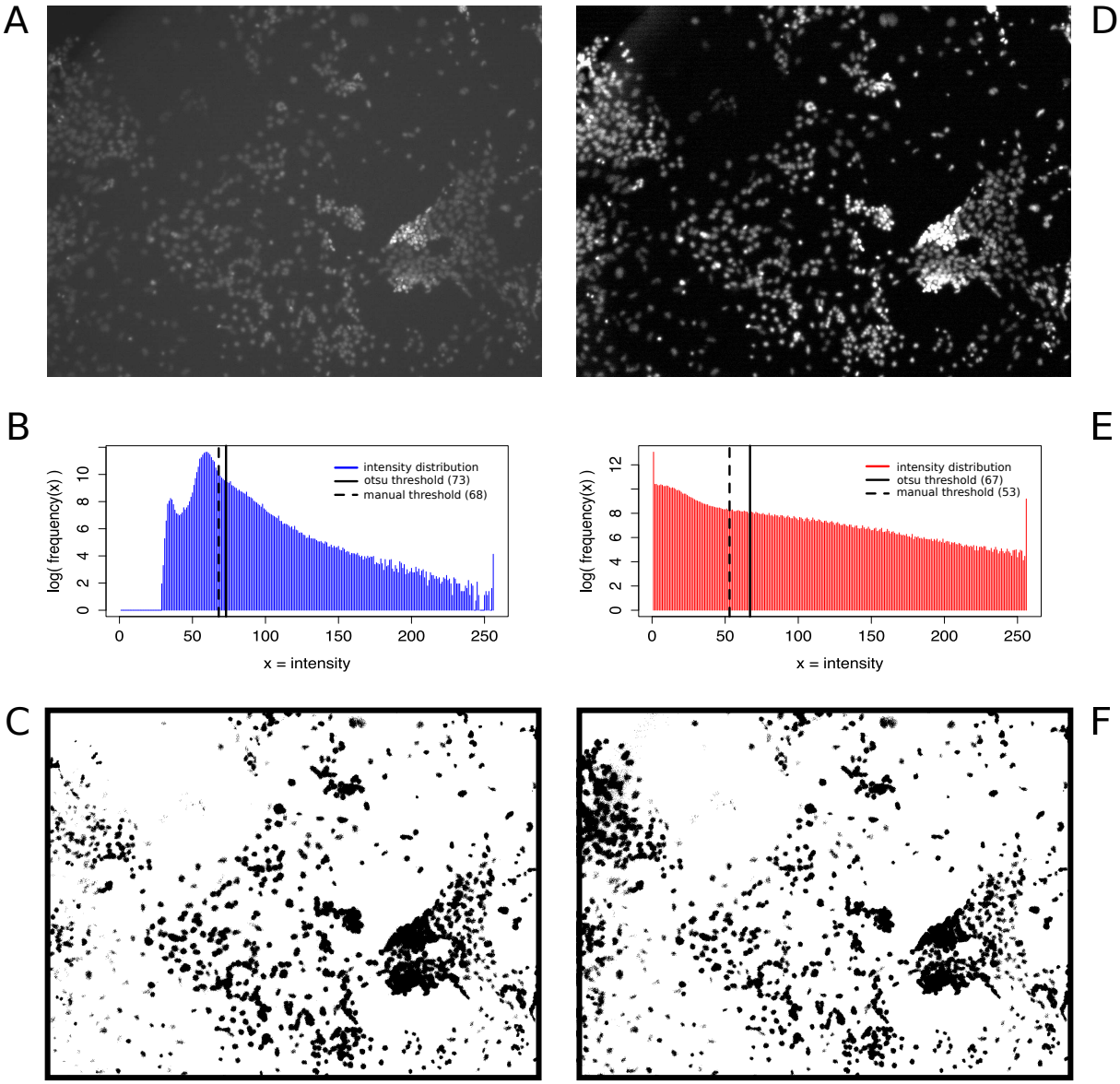

Figure 12: *(**A**) shows a typical raw fluorescence image. its background is not completely dark, it is uneven illuminated and shows vignetting (darker pixels at the edges) in the upper left corner. In (**D**), the same image is corrected by subtracting a previously calculated background. The cells show a much higher contrast against the background, than in (**A**). The intensity distribution of the corrected image (**E**) shows two clearly distinctive peaks at zero and 255 and a smooth distribution in between, whereas the raw image intensity distribution (**B**) obviously shows a much brighter image over all and an somehow unbalanced ratio between fore- and background pixels. The vertical lines show possible segmentation thresholds, that try to differ between pixels that belong to fore- or background. The solid line marks the threshold calculated by Otsu's method (see Section 2.4.3) and the dashed line a manually selected threshold. The segmentations, resulting from Otsu's threshold (**C**)(**F**) obviously show the improvement, a background correction has for the segmentation. Whereas (**F**) shows nearly all cells, (**C**) misses or shows a lot of them only partially segmented. All in all background calculation and correction really improves further processing and analysis, for example even rather simple methods, as a threshold for segmentation.*

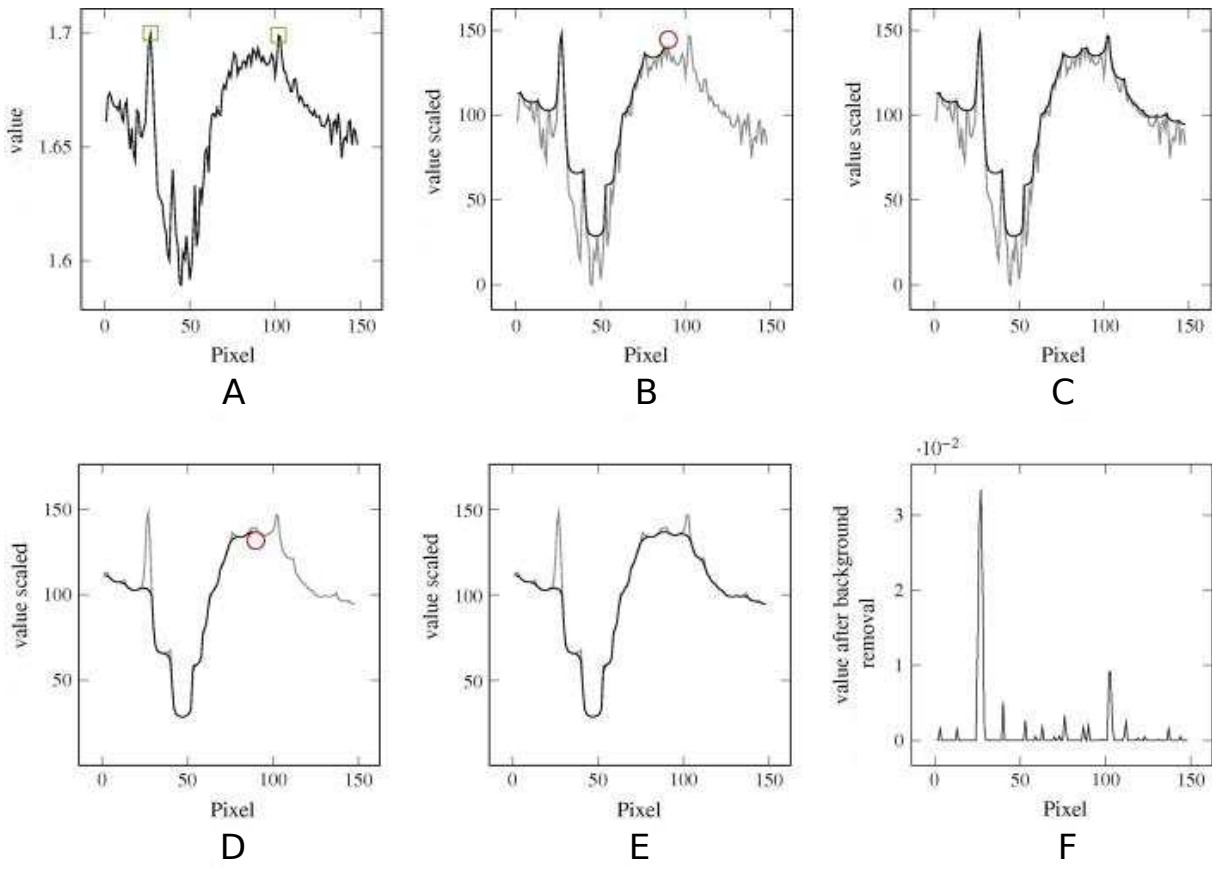

Figure 13: *A raw image (visualised as one-dimensional pixel vector in **(A)**) can be estimated, by visually using a ball rolling over and under the surface of the pixels intensities. By firstly rolling the ball over the surface **(B)**, the peaks of the intensities are captured very well, the more or less noisy background is smoothed at the pixel's maxima. After calculation of this surface **(C)**, the real intensities are truncated, and the ball is rolled under the surface **(D)**. Now one estimates the background, as the ball simply does not fit into thin peaks. After calculation of the background **(E)**, it can be subtracted from the previously calculated surface **(C)**, resulting in a corrected image **(F)**. The background is even and normalised to zero. The foreground objects clearly stand out and can further be used for segmentation and quantification more easily. (image adapted from Usamentiaga et al. [58])*

.

3x3 window for every pixel after the estimation. By checking the disable smoothing option, one can decide not to smooth the result.

Fluorescence Image Normalisation [47] This background calculation is one of the first step of a fluorescence image normalisation. Here we use a method previously described [47, 46]. So, it is especially suitable to images, showing a lot of background area and some blobs which are clearly separable. The algorithm (see Figure 14) firstly divides the image in tiles. These tile intensity distributions are measured separately and then clustered based on distribution features into two groups: one presenting all tiles containing a foreground object and the other containing all tiles, which seem to show only background. The background tiles are used to first interpolate a background by filling the holes, which originate from foreground tile exclusion. To ensure, that the background is spanning over the complete image, it is extrapolated to the borders and then ready to use for background correction.

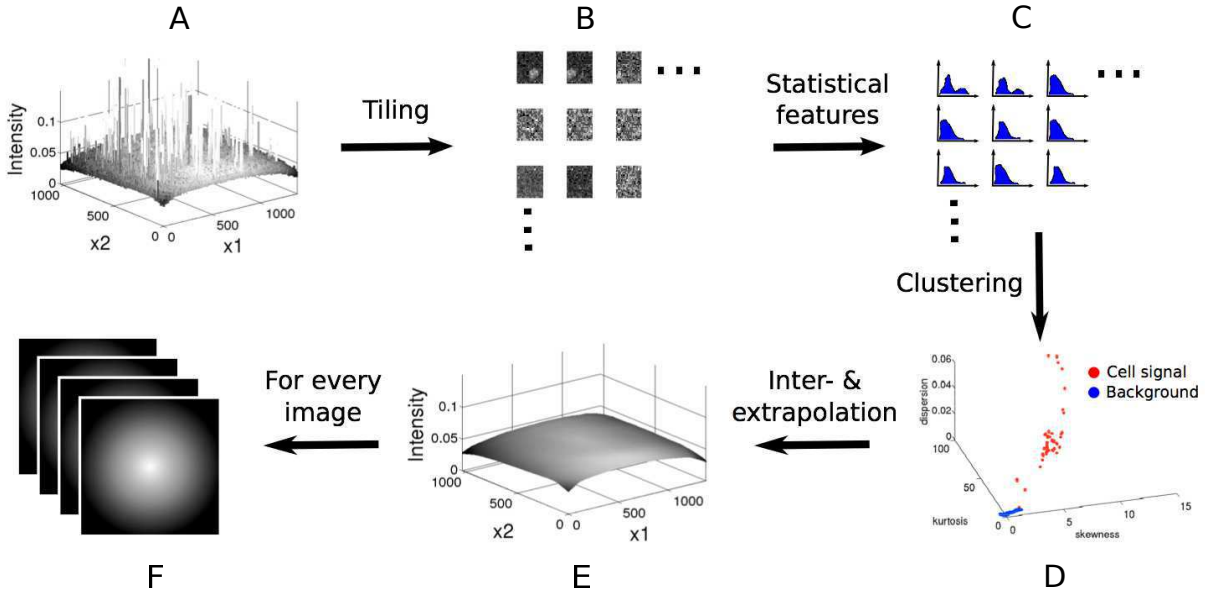

Figure 14: *An example image is visualised in **(A)** in 3D-representation with the pixels intensities plotted on the Z-axis. After breaking the image into overlapping tiles **(B)**, one can calculate statistical features for every tile **(C)**. Clustering selects tiles, which show only background **(D)**. Using these tiles, one can inter- and extrapolate a complete background image **(E)**. This can be performed for all pictures, for example of a time series **(F)**. These backgrounds can be used to correct the raw images directly or to calculate a so called gain image (see Figure 15) which additionally improves correction quality. (image adapted from Schwarzfischer et al. [47])*

Additionally to the background images, a so called time-independent gain image is calculated (Figure 15), to also be able to correct errors which are present in all images of a time line, due to experimental biases, like wrong illumination or different light permeability of the medium. Therefore a linear regression is performed on points virtually plotted as specific pixels of the background intensity versus the mean background intensity. The slope of the fitted line represents the pixels' gain value, like presented in Figure 15.

2.4.2 Correction

The correction of the raw image strongly depends on the calculated background. We therefore provided several possibilities, explained in the following:

Add Addition of the background to the raw image can be used in some special cases. The background has to show higher values as the foreground objects, for this method to make sense.

Subtract The background image is simply subtracted from the raw image. This method can be used when background pixels have low and foreground objects high values, which is the case most commonly.

Multiply Multiplication of the raw image with the background image can be useful in some special cases. The background has to be prepared in a special way, as the pixel values will otherwise be very high and hardly useful for further steps.

Divide One has to consider, that most resulting values from a division of the raw image

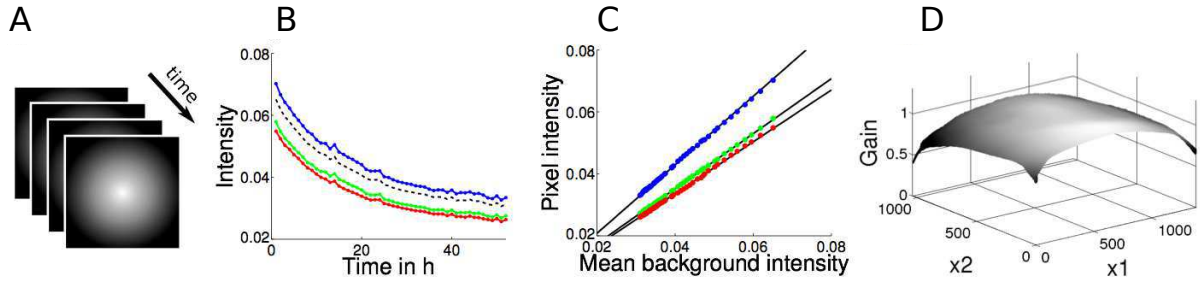

Figure 15: *To calculate the time-independent gain, the previously calculated background images of a time series (see Figure 14* ***(F])*** *are used* ***(A)****.* ***(B)*** *shows intensities of three specific pixels, plotted over time, whereas the blue point represents the center of the image, green the center point of the left border, red the point the upper left corner and black the mean intensity.* ***(C)*** *depicts the intensities, also plotted in* ***(B)****, over the mean intensities, as well as their linear regressions. The linear regression's slopes (of* ***(C)****) are then used as gain pixel values. By calculating this slope for every pixel, the gain image* ***(D)*** *can be constructed. The backgrounds as well as the gain images can be used to normalise the raw images as postulated in the Subtract and Gain correction method (see Figure 16). (image adapted from Schwarzfischer et al. [47])*

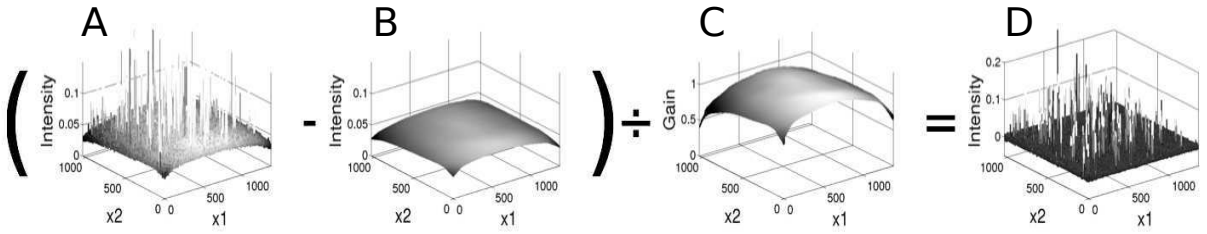

Figure 16: *Visualisation of the Subtract And Gain correction method with images calculated by the Fluorescence Image Normalisation (see Figure 14 and Figure 15). The background image* ***(B)*** *is subtracted from the raw image* ***(A)*** *and the result of this operation then divided through the gain image* ***(C)****. The normalised image* ***(D)*** *shows a flat background, which intensities are distributed around zero, and cellular signals, which set themselves clearly apart as high peaks. (image adapted from Schwarzfischer et al. [47])*

by the background image are distributed around one, not zero, as the background is commonly somehow estimated to fit the majority of the raw image's pixels.

Divide Minus One By subtracting one from the division of the raw image through the background image, the result is normalised to zero.

Subtract And Gain [47] This correction is specialised for time resolved background images, resulting from the fluorescence image normalisation, mentioned above. As there is a gain normalisation image calculated over time, one can subtract the background image from the raw image and divide the result by the gain image, as shown in Figure 16. For the single cell quantification tool, the gain image has to be located in the file system, as specified in the *TTT* experiment folder structure.

2.4.3 Segmentation

As the computer can not simply distinguish objects from background, one has to perform some steps to detect these objects. *ImageJ/Fiji* provides some of these, another more sophisticated, called Maximally Stable Extremal Regions [34, 21], was developed and

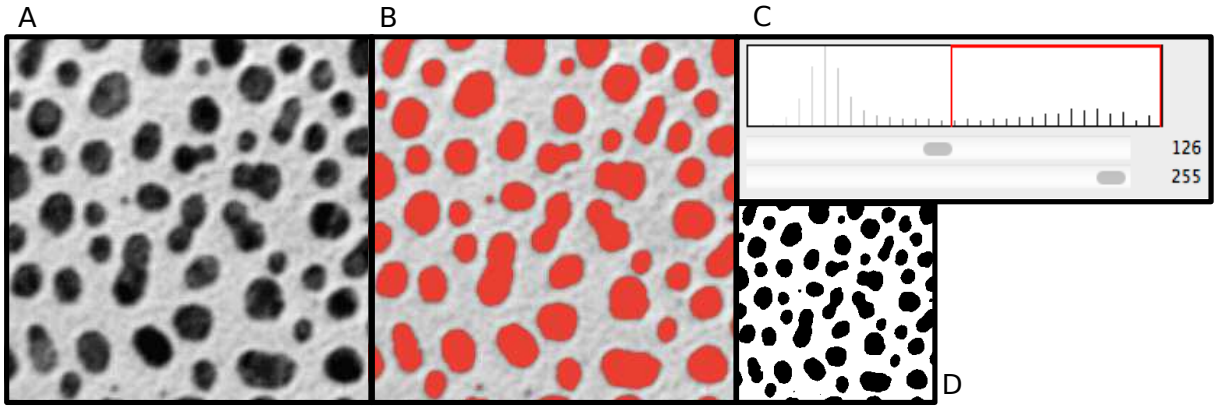

Figure 17: *An exemplary bright field image* ***(A)*** *shows some dark cells on a bright background. The histogram over its pixels is shown in* ***(C)****, as well as a manual threshold between 126 and 255 in red, meaning, that pixels lying in this range, are marked in* ***(B)****. After calculating the binary mask, based on this threshold, the resulting image* ***(D)*** *only shows bright (background) and dark (foreground) pixels.*

included on our own. The most important methods are presented below. The output of the segmentation methods is always a binary image, the so called segmentation mask. For the mask, *ImageJ/Fiji* uses white as background, and black as foreground color. For further steps, the objects have to be separated individually from the mask first. This task is performed by *ImageJ/Fiji* automatically on demand.

Manual Threshold As image intensities are distributed over a certain range (0-255 in 8-Bit grayscale), one can assume, that for example in fluorescence images all pixels, showing high intensities, belong to an object, against what pixels with low intensity belong to the background. This distribution of the pixels can be visualised as a bimodal histogram (see Figure 17), in the best case. By defining a threshold between the two groups, one can make the software to differentiate between foreground and background pixels. The more distinctive the two modes are in the histogram, the more clear is the result of the right thresholding. As shown in Figure 12, even for images, with no clear bimodal intensity distribution, a thresholding method can achieve good results. Clearly, thresholding is much more promising, if the uneven background was normalised before.

Automatic Threshold There are several methods to determine the best threshold automatically. Some of them are already included in *ImageJ/Fiji*, like the most familiar one, called *Otsu's* thresholding [39]. First *Otsu's* method defines two classes of pixels: one containing pixels below the threshold ($_1$), the other ($_2$) containing pixels over the threshold. Next, the method iterates over all possible thresholds (defined as t), maximising the difference between the two classes, the so called inter-class variance and minimising the intra-class variance. Therefore the class probability of the two classes is calculated for every threshold. $p(i)$ is the probability of a certain pixel, indexed as i, $x(i)$ its intensity value. The class probabilities are the sum of

all probabilities in the specific class:

$$\omega_1(t) = \sum_{i=0}^{t} p(i) \tag{2.4.1}$$

$$\omega_2(t) = \sum_{i=t}^{255} p(i) \tag{2.4.2}$$

The class means can easily be calculated by dividing the weighted class probabilities by the unweighted class probabilities. The class probabilities are therefore weighted with the pixel's intensities:

$$\mu_1(t) = [\sum_{i=0}^{t} p(i)x(i)]/\omega_1 \tag{2.4.3}$$

$$\mu_2(t) = [\sum_{i=t}^{255} p(i)x(i)]/\omega_2 \tag{2.4.4}$$

Then the inter-class variance can be derived:

$$\delta_b^2(t) = \omega_1(t)\omega_2(t)[\mu_1(t) - \mu_2(t)]^2 \tag{2.4.5}$$

And finally, *Otsu's* threshold value can be found, by calculating:

$$argmax_t(\delta_b^2(t)) \tag{2.4.6}$$

Maximally Stable Extremal Regions (*MSER*) The *MSER* segmentation is a more sophisticated threshold method. It iterates over all possible thresholds, trying to find different local thresholds, that fit some given parameters and reveal objects, which are clearly distinctive from the background. Therefore, the algorithm iterates over all thresholds, while storing all regions, that can be found at the certain thresholds. A region is a coherent set of pixels, meaning, that in the first iteration step, the whole image is detected and stored as a region. The further the iteration runs, the more regions are found, and the smaller they are. All new regions are associated with the regions of the previous iteration step. So, the algorithm builds a tree from bottom to top. After the tree was generated, the methods starts to look for regions of minimal size from top to bottom, which do not exist in the list of regions of the previous threshold. These minimal regions, so called seeds, then grow, during further iteration from top to bottom. Whenever a region remains the same during a certain period of iterations, it is called stable region and marked as such. The specific period of iterations is one of *MSER*'s parameters, called delta. After reaching the bottom of the tree, all found regions are filtered for another parameters, defining the minimal and maximal size of regions, to be found. Then, overlapping regions are identified, and those with the maximal delta selected, further being handled as maximally stable regions. These maximally stable regions and the regions, that do not have an overlap (and are therefore trivial maximal) are finally returned as foreground objects. The difference and great advance against the *Otsu's* thresholding is, that regions can be detected properly with different thresholds, which increases the quality of segmentation.

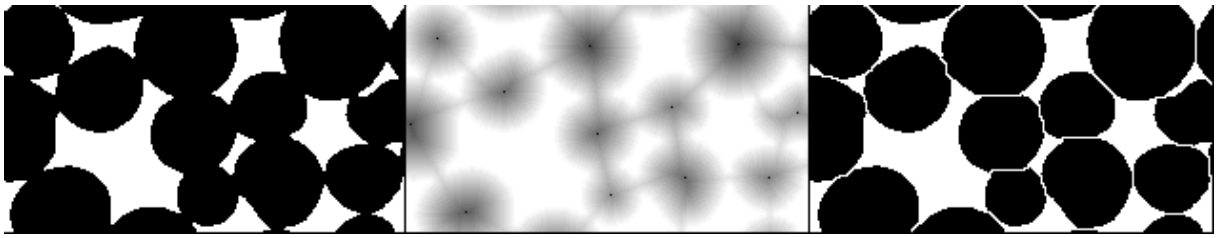

Figure 18: *Process of Watershed from left to right: raw binary image for example after segmentation, calculated euclidean distance map and marked ultimate eroded points and extrema of euclidean distance map, and result after complete watershedding (image adapted from Health [19]). Watershedding allows separation of clumped cells in binary images.*

Watershed [19] Watershedding (see Figure 18) is not a segmentation method, the way it is used in *ImageJ/Fiji*. Nevertheless, it can help improving the quality of segmentation significantly and, therefore, is mentioned here. By watershedding, clumped objects of binary images can be separated. Therefore, *SCQMI* calculates first an euclidean distance map and finds ultimate eroded points. The euclidean distance map is an image, which foreground pixels are coloured in a gray intensity, corresponding to the distance to the nearest background pixel. The ultimate eroded points are the centers of the objects that would be segmented by thresholding. Then, these ultimate eroded points or extrema of the euclidean distance map are dilated, until they touch a dilation border of another extrema or the border of the object. Watershedding can be used effectively on nearly every binary image, representing a segmentation.

2.4.4 Quantification

For quantification, one has to select the measurements, to be performed on the images' selections. Besides some structural measurements, like size, diameter, position and centroid, the intensity is measured in several ways. Therefore the sum, mean, median and deviation are calculated amongst others. One can simply choose a subset of all available measurements as parameters in the graphical user interface. Additionally, the file format of the results can be defined.

2.5 Bulk Execution

As `Workflow` stores the pipeline in its own object based structure, the workflow has to be exported as macro for *ImageJ/Fiji*, before it can be used further. Additionally the single steps of the workflow, especially of one with multiple images, have to be ordered, considering possible dependencies between the images. Therefore, we developed a special Petri-net (see Figure 19), which orders the workflow steps and exports them as textual macro.

Before being executed, the `Workflow` object, which was generated in background during workflow generation, is automatically converted into a Petri-net. A Petri-net consists of exactly two different node types: states and transitions. All connections between the nodes are directed, and only nodes of different kind can be connected. The transitions

represent the pipeline actions, can be scored, and are initialised with a score of zero. The states represent the images, which are input and output of the pipeline steps. The Petri-net is defined as unsolved, as long as two transitions have the same score. To solve the net, first all sinks and sources are identified. Then the transitions are scored recursively, starting with those transitions, that lead to sinks, and a score of 1. The more steps the scoring recursion has already taken, the higher the next transition's score is increased. At the end, the transition with the highest score is the first, which has to be performed in the macro. All following actions can simply be identified by ordering them, according to their score. The Petri-net, representing the `Workflow` object, is solved automatically in background, converted into a simple textual representation of the correctly ordered macro, and then passed to the bulk executor, which performs it on all images, which were previously selected.

As a workflow can contain more than one picture's pipeline, the ordering of the pipelines themselves has to be solved, first, by transforming the pipelines into states and all actions between them into transitions (Figure 19, ***(B, iv)***). Afterwards, the single pipelines can be converted into Petri-nets ***(III)***, solved one after the other ***(IV)***, and finally assembled to an complete list of workflow steps, using the information of the Petri-net, described by the single image pipelines.

A Petri-net can be unsolvable, for example, if it contains cyclic dependencies between intermediate images or parts of the workflow. As these Petri-nets would lead to infinite loops, Petri-net solution is automatically observed, stopped in case of infinite calculations and an error reported. The error can be corrected, using the workflow creation dialogue and executed again.

As described in Section 2.3, *SCQMI* was thought to be capable of executing the workflow in parallel locally as well as on a grid. The `BulkExecution` class provides the interface to handle these options and the `VirtualData` window already shows the disabled combo boxes, which will enable the user to select these execution modes. As the implementation of those modes was well out of the scope of this work, the macro can only be executed sequentially on the local machine in automatic mode.

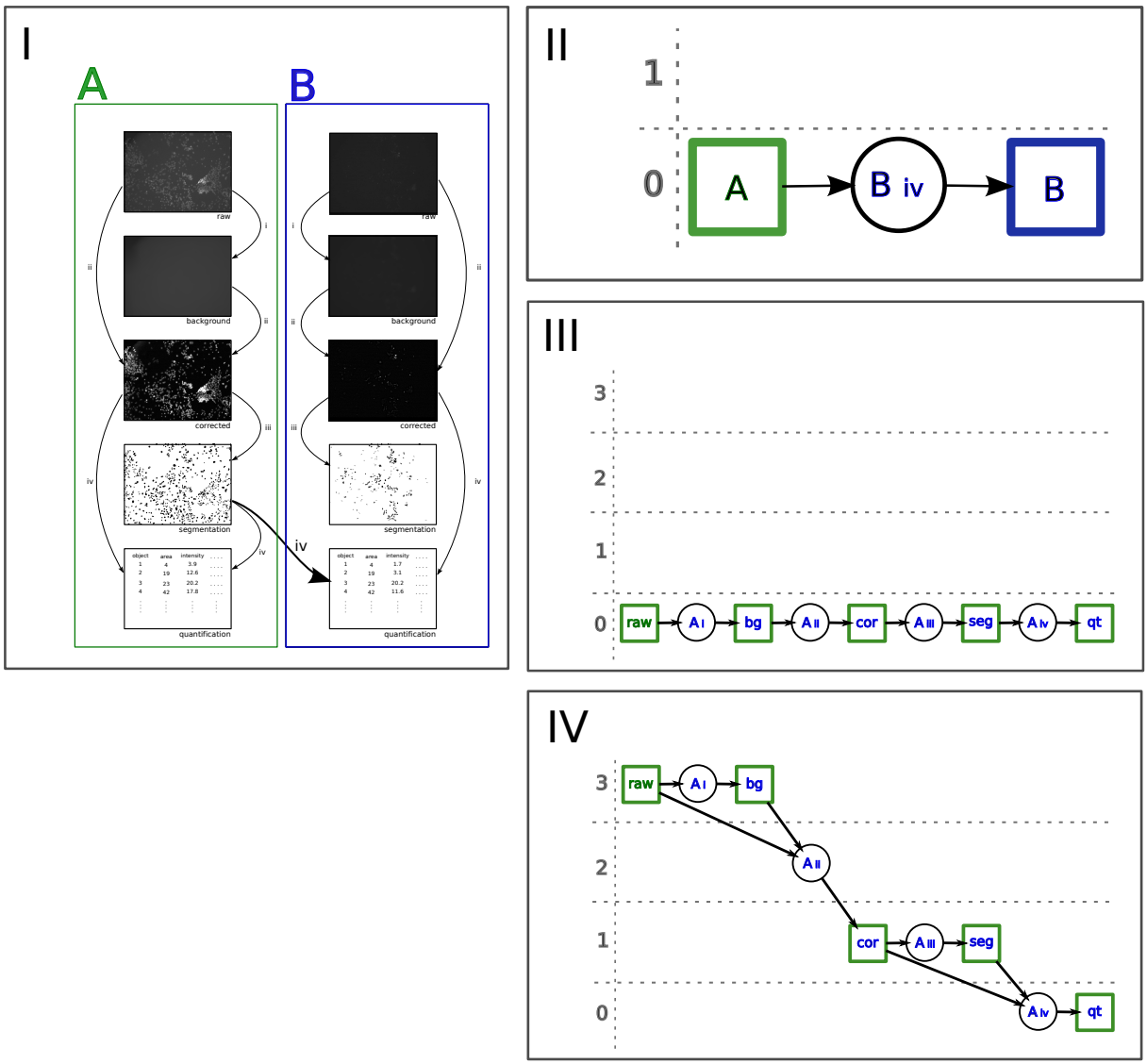

Figure 19: *A typical workflow (**(I)**, also see Figure 7), consisting of two picture's **(A)(B)** pipelines, with actions between them, has to be performed in the right order. Otherwise, some intermediate image dependencies could be missing. To prevent this case, all image pipelines (if there are multiple ones) are translated into states of a Petri-net **(II)**, being linked, by transitions, which represent the actions between them, first. Here, states are visualised as rectangles, transitions as circles. Another Petri-net is being build for every image pipeline itself, whereas intermediate images are represented by states and the four standard actions (background calculation **(bg)**, background correction **(cor)**, segmentation **(seg)**, quantifications **(qt)**) as transitions (exemplary shown in Frame **(III)** for image pipeline **(A)**). All those nets are then solved **(II)(IV)**, by recursively increasing the score of the transitions (depicted as certain levels). The net, showed in **(II)**, was already solved trivial after creation, as there are too less transitions. Frame **(IV)** shows the solution of the net in frame **(III)** . Obviously, the right order of the image's pipelines, can be directly read from the solved Petri-nets and assembled to the whole workflow, regarding the results of the superior Petri-net.*

3 Validation and Evaluation

To ensure, that the developed tool (*SCQMI*) performs as expected, we validated the single steps and evaluated differences, if needed. As a tool for single cell quantification was already prototyped in *MATLAB* (called *sQTFy* in the following), we took the intermediate outputs of this tool as reference for some validations.

3.1 Background Calculation

As the *Fluorescence Image Normalisation* method, described in Section 2.4.1, was not yet implemented as a plugin for *ImageJ/Fiji* and this task was well out of scope of this work, we further calculated the backgrounds with this method's implementation in *MATLAB*. Therefore, there was no need to validate a background calculation. The method itself was properly validated against others, by imaging beads and a fluorescein solution and validating the images after background calculation, correction and segmentation [47].

3.2 Correction

Although, the correction steps seem to be rather easy, we had to check whether the type conversions influence the correction results. Also, we had to check whether *ImageJ/Fiji* allows negative pixel values, as otherwise the results of the developed tool (*SCQMI*) would show a difference to *sQTFy*. This difference had to be evaluated properly. Therefore, we used an image from a real experiment and a pre-calculated background image. The raw image was divided by the background image and one subtracted from the result in *MATLAB* as well as in *ImageJ/Fiji*. This procedure is exemplary shown in Figure 20. The results were compared, as shown in Figure 21. Although the results are not equal, they still do not show noteworthy differences. Obviously, all variance is very low ($correlation \approxeq 1$), pointing to rounding errors.

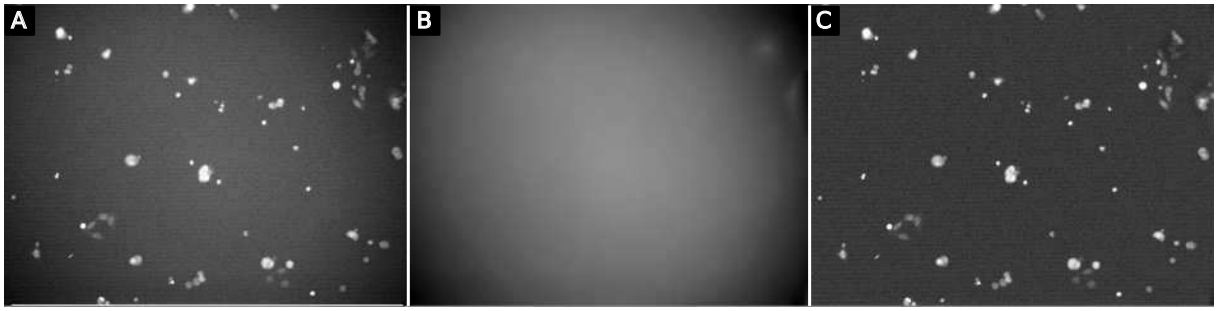

Figure 20: *For validation and quantification of possible differences of image correction, a raw image* ***(A)*** *was divided by a background image* ***(B)****. The resulting image* ***(C)*** *was compared between ImageJ/Fiji and MATLAB (see Figure 21). Obviously, the quality of the result image improved much against the raw image, regarding the equality of background pixels and the contrast between fore- and background. As shown in Figure 12, depending on the quality of the background image, this image correction enables much better results of further image processing and analysis.*

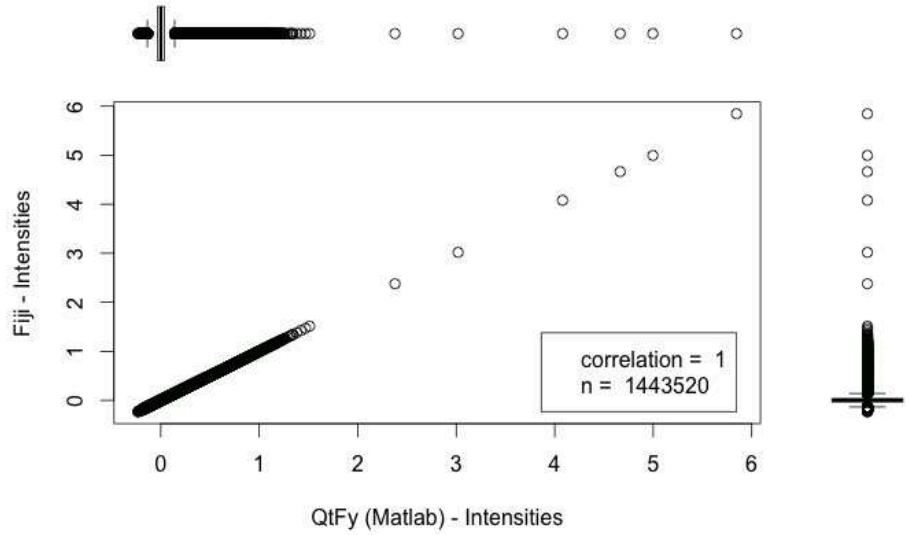

Figure 21: *The values of pixels of the corrected image strongly correlate, as they have hardly any notable differences, and show a straight line in scatterplot. The boxplot shows, that the majority of pixels is distributed around zero, enabling negative values, in both cases. This improves detection of foreground objects, for example by setting an intensity threshold.*

3.3 Segmentation

Segmentation is always a crucial part of image processing, wherefore we evaluated two examples. As the *MSER* algorithm was used in both *MATLAB*, for prototyping, and *ImageJ/Fiji*, we shortly checked, whether the outputs of the basic algorithm are equal, using the same parameters. After acknowledgement of equality, we further checked the correlation of the results of the methods wrapping *MSER*. These wrappers create the binary image from the so called seeds and associated thresholds, calculated in the algorithm, as explained in Section 2.4.3. Additionally, holes are filled. As reference we segmented both images manually, by using *ImageJ/Fiji*'s selection tool and selection manager. These references were then compared to various segmentations and their post-processes. We used ***(i)*** the wrapped *MSER* algorithm of *ImageJ/Fiji*, ***(ii)*** the previous with an additional watershedding, ***(iii)*** the *MSER* algorithm of *sQTFy*, and ***(iv)*** the semi-automatic method, also used in *sQTFy*. The semi-automatic method of *sQTFy* uses the *MSER* algorithm to identify the center of objects, and then lets the user segment the object, by performing an *Otsu* thresholding on a small aperture, centered on the *MSER*'s seed.

We used the Normalised Sum of Distances (*NSD*) [9, 8] to score the single objects, detected by the various methods. As the NSD is a spatially aware scoring method of two sets, it is a weighted measure, that takes the distance into account, non-overlapping pixels have to the reference. Points, that have a high distance, degrade the score more, than more points with less distance. To calculate the NSD, all identified objects of two images have to be associated pairwise. This was performed by a symmetric nearest neighbour calculation: for every object in the first image (R), an object of the second image (S) was searched by minimising the distance between their centers. To consider missing or additional objects, the same was done for all objects of the second image. Only if the same pair was proposed from both, the two objects were associated as the same one in two different images. Next, the *NSD* between al these objects was calculated in both directions, as this score is not

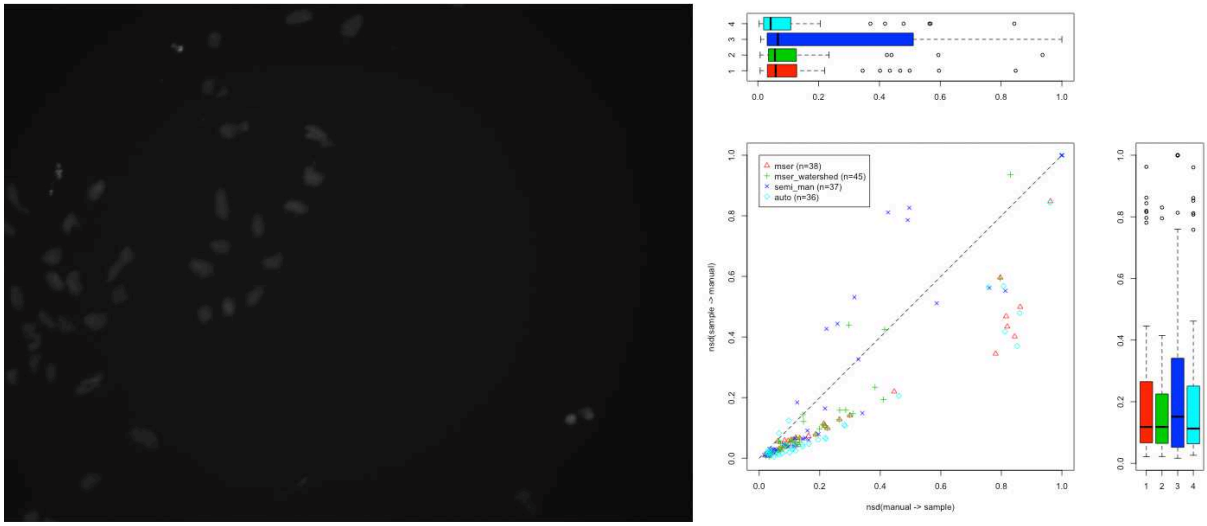

Figure 22: *Validation of an image with few, clearly distinct cells* ***(left)****, showed the quality of various segmentation methods:****(i)*** *ImageJ/Fiji's MSER (red,* $n = 38$*),* ***(ii)*** *MSER and Watershed in ImageJ/Fiji (green,* $n = 45$*),* ***(iii)*** *MATLAB's MSER (cyan,* $n = 36$*), and* ***(iv)*** *the semi-automatic methods of sQTFy (blue,* $n = 37$*). For comparison of the methods, the identified objects were compared, using the NSD, and plotted. Both results of the asymmetric NSD were plotted on the axis of the scatterplot and additionally visualised in the boxplots above and right of the scatterplot. Except the semi-automatic method from sQTFy, all methods show the same good quality of segmentations, as the majority of NSDs is located between 0.0 and 0.2. Some exceptional points, located between 0.2 and 0.8, point to over- and under-segmented cells. Values equal one of the semi-automatic method* ***(iv)*** *can be explained by the method itself: the MSER might find some seeds in the image, but in the aperture, centered over these seeds, Otsu's thresholding can not reveal the right cells, but for example artefacts or neighbouring cells.*

symmetric.

$$NSD_{R,S} = \frac{\sum_i \|R_i \neq S_i\| D_i}{\sum_i D_i} \tag{3.3.1}$$

The *NSD* between two objects, which are defined as sets of pixels, R and S, is calculated by defining $R's$ border first. Then, one iterates over all pixels of the union $R \cup S$, calculating the current pixel's i distance D_i to the border and sums the distance, if the pixel is only member of one set R or S. To get the *NSD*, this sum is divided by the sum over all distances. Obviously, the two objects absolutely match, if the *NSD* equals zero and do not overlap, if the score equals one.

The first of the validated cases (see Figure 22) is an endpoint staining picture with only few, clearly separated cells. We chose $delta = 1$, $min\ size = 50$ and $max\ size = 2000$ and as parameters for this case. For the second case (see Figure 23), we chose a much more crowded picture with a lot of clumped cells and therefore set the parameters $delta = 1$, $min\ size = 30$ and $max\ size = 450$.

Figure 22 shows, that the majority of calculated *NSD*'s, between the method's results and the manual segmentation, is lower than 0.2, indicating very good matches. Some values between 0.2 and 1.0 indicate over- or under-segmentations. Values equal 1.0, like present in the blue case, indicate wrong association. As the method, presented by blue points, identifies objects by *MSER* and segments them by a manual threshold, the errors most likely occur during thresholding. The center, which is later used for association with ob-

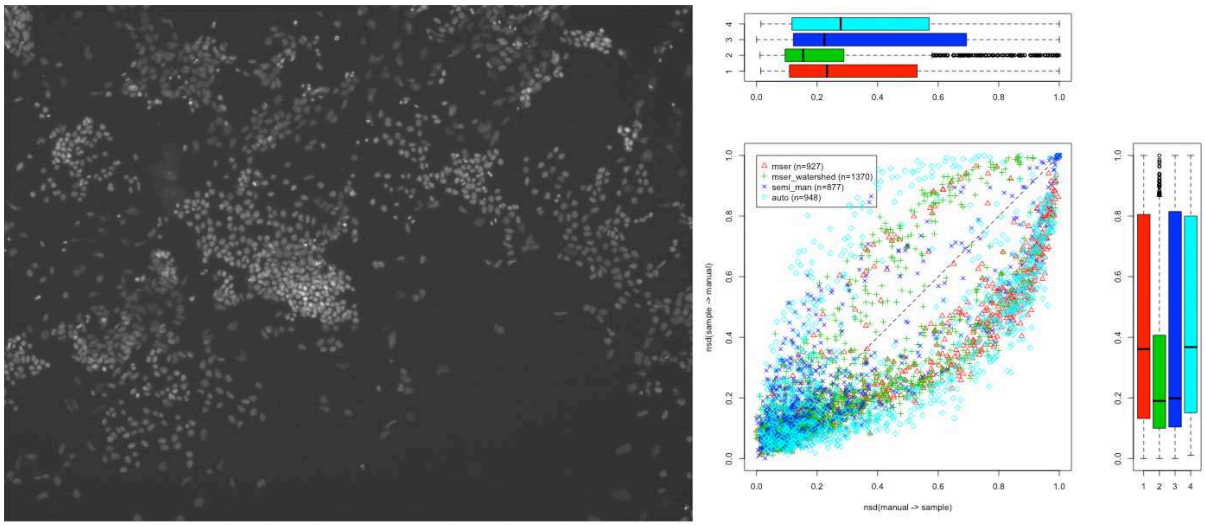

Figure 23: *Validation of a more realistic image* ***(left)****, as in Figure 22, revealed a sample quality of different segmentation methods, like* ***(iii)*** *MSER implemented in MATLAB (cyan,* $n = 948$*) and* ***(i)*** *ImageJ/Fiji (red,* $n = 927$*), the* ***(iv)*** *semi-manual method used in sQTFy (blue,* $n = 877$*), and* ***(ii)*** *MSER followed by a watershedding in ImageJ/Fiji (green,* $n = 1370$*), under realistic conditions. The image contains much more cells, which are partly clumped, very differently illuminated, and of different size. The plot on the right shows the NSDs of the segmentation results as a scatterplot and as boxplots. As the NSD is an asymmetric score, both results were plotted on different axis and additionally visualised in different boxplots respectively. The majority of NSD values remains in a range between 0.0 and 0.2 for the method, which combines MSER and watershedding. The other methods produce values, that are distributed over a much bigger range, pointing to worse results. Especially the large group of blue dots (****(iv)****, semi-automatic method of sQTFy), located around one, shows, that the problem with seed-region associations remains in real-experiment images.*

jects of other images, is already fixed after performing *MSER*. Using Otsu's thresholding, one might select the wrong object near the real center, as the real object is not as easy to detect as the wrong one. The other three methods show the same quality.

As the image, used in the first validation, is hardly one, that occurs very often, we selected another one with much more cells, which are partially clumped, and not that constantly illuminated. Figure 23 shows values, that are obviously higher than in Figure 22. Especially the results of *MSER* in *ImageJ/Fiji* (red, $n = 927$) and *MATLAB* (cyan, $n = 948$) are worse. Only *MSER* in *ImageJ/Fiji* and watershedding (green, $n = 1370$) lead to nearly the same good results. Also the values of the semi-automatic method of *MATLAB* show the same distribution like in Figure 22. Especially the wrong seed-region associations influence this result, as there are a lot of points equal one. The advantage of *ImageJ/Fiji*'s *MSER* followed by watershedding is, that clumped cells can be distinguished even after the segmentation. This leads to a clearly better result, than without watershedding.

3.4 Runtime Issues

The user should not have to spend too much time in creating the workflow. Therefore, we minded to implement a graphical user interface, which is as intuitive and user-friendly as possible. After some time of familiarisation with *SCQMI*, most users are able to create

and process a standard workflow, as depicted in Figure 7, in less then five minutes. Also the runtime of this workflow on images keeps itself in limit. On a standard laptop (Intel Core i5, 1.8 GHz) it took about 10 minutes to calculate the workflow on 60 images.

4 Application

We developed a tool (called *SCQMI*), capable of processing time lapse movies in very specific manners, defined in workflows (Section 2). These workflows can be created on single exemplary images and performed on a stack of other images. A standard pipeline consists of four steps: background calculation, correction, segmentation and quantification. We used *SCQMI* on a real dataset. In the following we motivate this investigation of embryonic stem cells, describe the experiment conditions, as well as the post-acquisition processes, and present and discuss the results.

4.1 Motivation

As briefly exposed in Section 1.1, embryonic stem cells (ESC) are of great interest, investigating pluripotency and differentiation in general, and, for example, cancer in special [44]. ESCs are the pluripotent inner mass cells, extracted from a blastocyst (see Figure 24). The blastocyst emerges from the totipotent morula, which is aggregated during the first rapid divisions of the zygote. The ESCs in this early embryonic stage have potential to build most other somatic cells, for example of the circulatory, nervous or immune system.

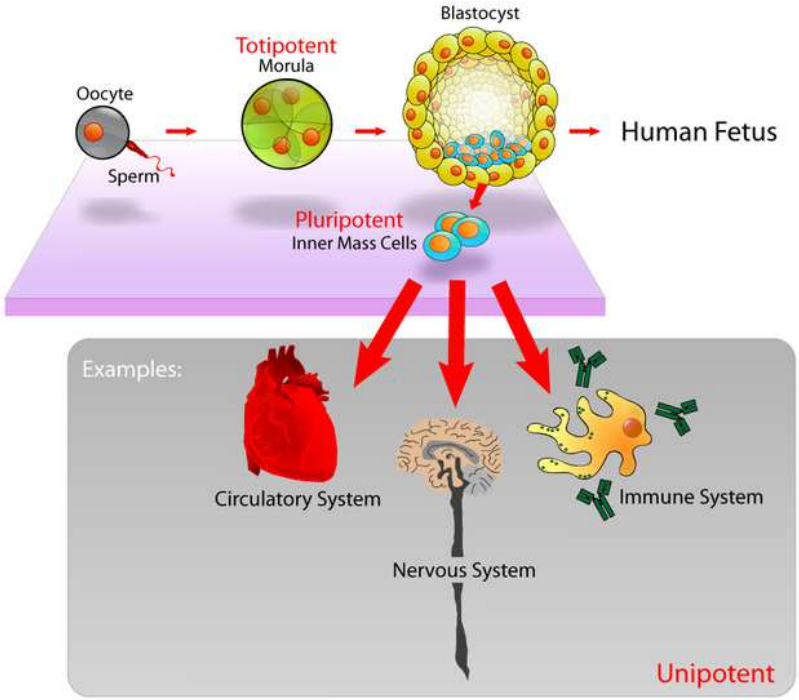

Figure 24: *A zygote divides rapidly multiple times after fertilisation. The aggregation of these divided cells is called morula. its single cells are totipotent, meaning, that they can differentiate to every somatic cell. From the totipotent morula emerges the blastocyst, containing pluripotent mass cells in the interior. These cells can be extracted and used as embryonic stem cells (ESC) for biological investigations of stem cells and the early development of organisms, as they are capable of emerging to nearly all somatic cell types, forming for example the circulatory, nervous or immune system. (image adapted from Jones [23])*

So, embryonic stem cells have characteristics, making them perfect suitable for in-vitro investigations [33]. On the one hand, they remain pluripotent under certain conditions. On the other hand, as long as they are not differentiated, they self renew, what facilitates biological investigation very much. ESCs are cultured under different conditions to achieve different behaviour [63]. On a medium, containing serum and LIF (leukemia inhibitory factor), the ESCs self-renew over many weeks, but also partially differentiate. To reduce the differentiation rate and keep the cells in a more stable pluripotent state, the ESCs can be cultured on a *3i* or *2i* medium, which contains three (or respectively two) inhibitors of various proteins in addition to serum, and drives the cell towards a so called ground state. By plating ESCs on a medium, only containing serum in absence of LIF, they begin to differentiate. Although the underlying processes are studied until more then three decades, they are still not completely understood. A lot of transcription factors were identified, being included in the process of regulating pluripotency and differentiation, in which Nanog, Oct4 (octamer-binding transcription factor 4) and Sox2 ((sex determining region Y)-box 2) are thought to be master regulators. Also Rex1 (Zinkfinger protein 42) and Klf4 (kruppel-like factor 4) are known to play a key role in the process of pluripotency.

Traditionally, the pluripotent state and differentiation of ESCs was defined by high absolute levels of some transcription factors. The levels of Nanog [6], Oct4 [61] andKlf4 [64], for example, are thought to define the current state of pluripotency or differentiation. As correlations between these levels had been observed, the transcription factors were suspected to auto-regulate themselves. Due to this assumption, the correlations between the pluripotency associated transcription factors were further investigated [27, 32]. Actually, the results (see Figure 25) evidenced, that the inquired transcription factors build a closed meshed regulatory network, in which they are regulating each other.

During detailed investigation of the pluripotency associated transcription factors Oct4 and Nanog [13], it was proposed, that not the absolute levels, but the ratio between the transcription factors drives ESCs towards specific states. Three states were therefore defined: pluripotent, lineage-primed and differentiating. Additionally grouping the cells into the lineage-primed group, was also able to explain a heterogeneity in gene expression of pluripotency transcription factors [2], meaning, that ESCs, which are and remain definitely pluripotent, show strongly divergent absolute gene expression levels. But as the ratio between the observed factors remained equal or at least alike, this behaviour, was explainable.

In scope of this work, we investigated the correlations between Nanog, Oct4, Sox2, Rex1, and Klf4. As we had access to images, with six instead of two wavelengths, we were able to correlate multiple factors on single cell level with high accuracy and to analyse partial correlations of multiple factors instead of binary ones. The experimental background and performed steps towards analysis are described in Section 2.1. Analysis and results are presented in Section 4.3.

4.2 Single Cell Imaging and Data Processing

Imaging was performed on mouse embryonic stem cells (mESC). Therefore, the ESCs were cultured in basal medium, charged with serum, LIF and other supplemental chemicals [17]. As ESCs have to express a fluorescence reporter, to be visible during time lapse fluorescence imaging, VENUS was knocked into the Nanog gene of the ESCs. VENUS

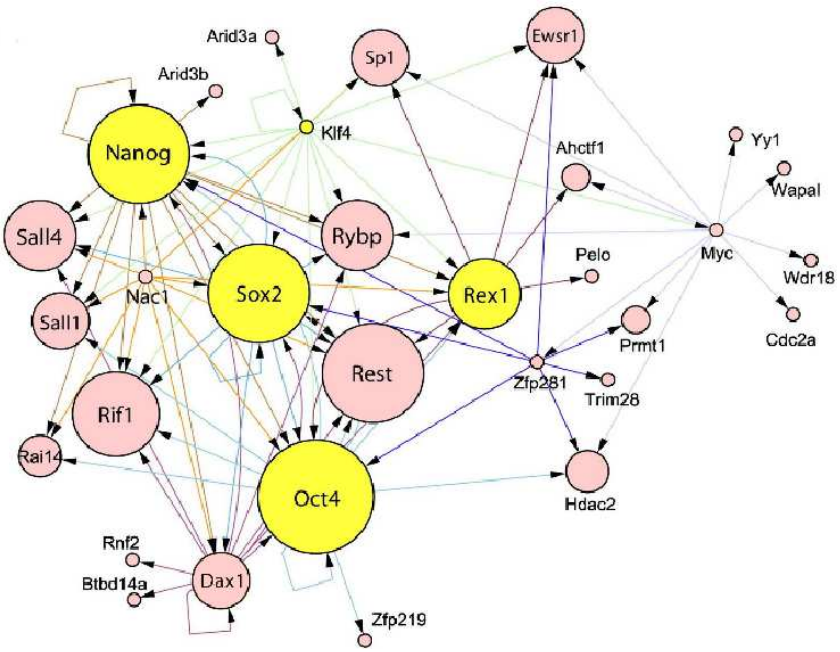

Figure 25: *The transcriptional regulatory network shows multiple factors within the protein interaction network associated with pluripotency. Regulations were identified, using in vivo biotinylation mediated chromatin immuno precipitation (bioChIP) and global target mapping (bioChIP-chip) [27]. Nodes represent targets of transcriptional regulation. Yellow circles represent the factors examined in scope of this work. The degree of factor co-occupancy of chromatin regions is visualised by circle size. Arrows indicate the direction of transcriptional regulation. (image adapted from Kim et al. [27] and modified)*

[38] is an alternative to green fluorescence protein (GFP), which is more stable in cell conditions and responds more quickly to stimulation. By knock-in fusion, the VENUS gene was directly linked to the Nanog gene, meaning, that they were expressed as one protein. The advantage of this linkage is, that different half life of reporter and target proteins are tackled, and quantifications of the reporter can directly be associated to quantifications of the target protein.

Using these knock-in fusion ESCs, time lapse imaging was performed at bright-field and VENUS wavelength. After transferring the cultures into 96 well slides, the channels were imaged every 30 minutes, using illumination times of 2 milliseconds for bright-field, 1000 milliseconds for the VENUS channel and 100 milliseconds for the immuno-fluorescence channels. After nearly 3 days the Zeiss Axio Observer Z1 microscope (Zeiss, Munich, Germany) had taken 134 time point images with a resolution of 1388x1040 pixels per position and wavelength. At the last time point (135) various proteins of interest were immuno-fluorescence imaged. For immuno-fluorescence imaging the cells were fixed on the medium and treated with immuno-fluorescence labelled dyes, specific to the target proteins of interest. Time lapse movies can not be continued after immuno-fluorescence staining, as the cells are not viable after fixation on the medium. Two different time lapse movies were thereby generated, differing only in the set of stained proteins in the last time point, whereas in both Nanog, Oct4, Klf4 and DAPI (4,6-diamidino-2-phenylindole) were used. DAPI binds persistently to DNA, especially A-T rich regions and can, therefore, be used as nuclear marker, which always shows nearly even fluorescence, as against the other fluorescence markers. As the amount of DNA increases during cell cycle, DAPI can also

be used as cell cycle marker. In first movie ***(I)***, spanning over 60 positions, additionally Rex1 was stained, in the second (31 positions, ***(II)***) Sox2.

The described imaging produced about 20 GB of data, spanning over 91 positions, 135 timepoints, and up to 6 wavelengths. The data was stored in TTT file hierarchy [37] (see Section 2.3).

To quantify the generated images, we, of course, used the previously developed *SCQMI*. Using the TTT import option (see Figure 4), we loaded the whole experiment into the underlying data-structure. Background images were already computed by an external tool, previously described in [47, 46], which performs the algorithm *Fluorescence Image Normalisation*, described in Section 2.4.1. These backgrounds were automatically associated with the imported raw images, as they also were saved in TTT file hierarchy, and added to the data-structure. The multiple dimensions were renamed quickly, to be recognised easier during subsequent workflow generation. As we did not want to analyse time resolved results, we build a workflow, which enabled us to quantify only the stained wavelengths in time point 135 and Nanog in time point 134 with a segmentation of DAPI images.

As the immuno-fluorescence dyes interfere with Nanog, the prior time point image was used for Nanog quantification. After immuno-staining, the cells are slightly shifted, due to the staining process. We therefore registrated the Nanog images on time points 134 and 135, by calculating and applying a rigid transformation matrix. This task was mastered by *elastix* [28]. The transformed Nanog image was further used as Nanog "endpoint" image.

Therefore, we selected an raw image of time point 135, showing DAPI, from the data preview window and started workflow generation. In the workflow, the background image had to be loaded from data-structure. The raw image was corrected by using the *Divide Minus One* method (see Section 2.4.2). The corrected image was on the one hand used for segmentation with the *MSER* method (Section 2.4.3) and on the other hand for quantification, using all parameters. The parameters of *MSER* were visually optimised during runtime, using the preview window. $Delta = 10$, $min\ size = 50$ and $max\ size = 2000$ showed the best results and were used for all segmentations. The quantification results of all possible measurements were, as well as all correction and segmentation images, saved in the TTT folder hierarchy. In addition to the DAPI image, the images of all other wavelengths of the same position were imported into the workflow. These were also background corrected with already precalculated backgrounds, but not segmented by *MSER*. For quantification, we used the corrected images and the DAPI segmentation, previously calculated. The results and intermediate images of those wavelengths were saved in the TTT hierarchy, too.

After generation of the workflow, all images were processed, by selecting the specific DAPI images as raw images and monitoring the failure free progress in the *workflow process monitor*. Afterwards the data had been saved into the TTT file hierarchy and was ready to be analysed.

Table 1: *In addition to the by-eye comparison of the similarity between the various distributions (see Figure 26), also the statistical comparability was checked. P-values for selected statistical equality tests of the distributions of measured transcription factor intensities of Nanog, Oct4, Rex1 or Sox2, Klf4, DAPI and the cells area were calculated. The Wilcoxon rank-sum test [60, 49] tests whether the medians of samples from two independent, not necessarily normal distributed variables are equal. The Kolmogorv-Smirnov test [51, 50], uses the supremum of the distances between the two cumulative distribution functions as statistic. H_0, the null hypothesis was in every case, that the two experiment's distributions of a specific feature are equal. As all p-values are far lower than the 0.05 significance level, the null hypothesis had to be rejected always, meaning that the distributions are not equal under the consideration of a 0.05 significance level.*

	Wilcoxon	Kolmogorov-Smirnov
area	$4.74e-09$	$< 2.2e-16$
Nanog	$1.08e-40$	$< 2.2e-16$
Oct4	$3.67e-86$	$< 2.2e-16$
Rex1/Sox2	$< 2.2e-16$	$< 2.2e-16$
Klf4	$< 2.2e-16$	$< 2.2e-16$
DAPI	$1.72e-142$	$< 2.2e-16$

4.3 Analysis and Results

As we quantified two different movies, one showing Rex1 (further referred to as movie ***(I)***), the other Sox2 (movie ***(II)***) in addition to Nanog, Oct4, Klf4 and DAPI, we also analysed their results separately and compared them to each other subsequently. Thanks to the automatic quantification, we could achieve big datasets, what makes signals, being found in the data, highly significant. In detail we were able to measure 22180 cells in movie ***(I)*** and 15678 cells in movie ***(II)***.

We first checked the comparability of the two independent experiments. Therefore, we firstly controlled the distributions of cell area and absolute intensities of the measured transcription factors Nanog, Oct4, Rex1, Sox2, Klf4 and DAPI by eye. As depicted in Figure 26, all pairwise plotted distributions show high similarity, except the measurements of Rex1 and Sox2 ***(D)***. Some minor differences between the distributions, may be explained by the experiments, which conditions and results can never be exactly equal. The assumption of the two experiment's data being very comparable, was affirmed by calculation of Pearson correlations between the measured features of the specific experiments (see Table 2 and Table 3). For example, the correlations between area and DAPI (both $r = 0.70$), area and Nanog ($r = 0.47$ ***(I)***, $r = 0.46$ ***(II)***), and Nanog and Oct4 ($r = 0.50$ ***(I)***, $r = 0.48$ ***(II)***) show strong analogy. Nevertheless, non-parametric statistical testing, whether the distributions are equal, revealed, that they are not, using the Wilcoxon rank-sum test [60, 49] and the Kolmogorov-Smirnov test [51, 50]. These non-parametric statistical tests are used to decide, whether two samples are drawn from the same distribution. In advance to other tests, they do not assume normally distributed data sets, what fits the demand of the distributions, presented in Figure 26. The rejection of the null hypothesis may be explained by the fact, that the results of different experiments, although they were performed under the same conditions, are never exactly equal. The precise reason for the difference between the two datasets, although far enough cells were observed, remains unknown and has to be revised in further analyses.

An overview over the absolute cell intensities and areas in movie ***(I)*** (see Figure 27),

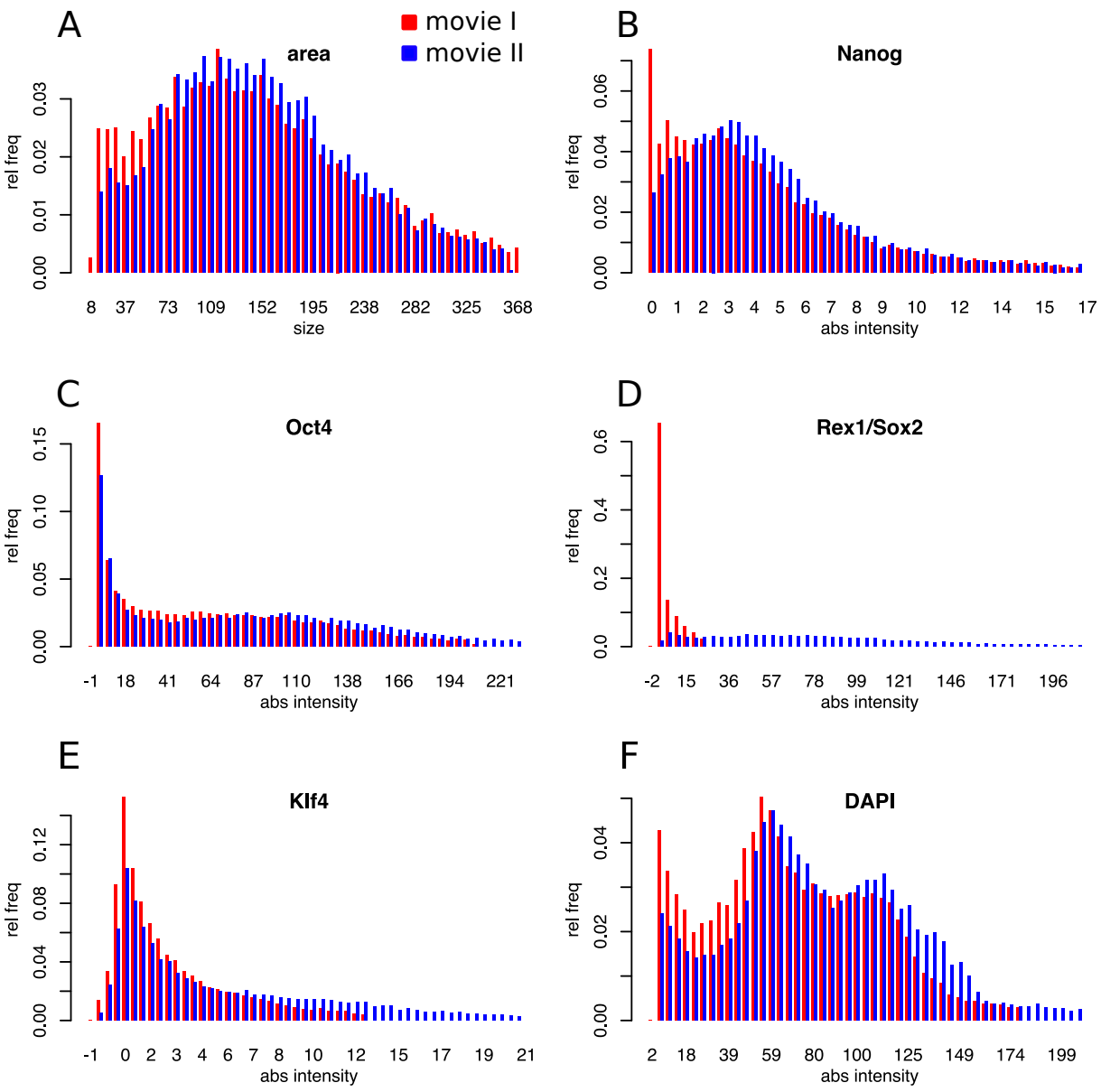

Figure 26: *Comparability of two quantified experiments (**(I)** for experiment incorporating Rex1 and **(II)** for experiment incorporation Sox2) was checked, regarding the distributions of cell area **(A)**, absolute intensity of Nanog **(B)**, Oct4 **(C)**, Rex1 or Sox2 **(D)**, Klf4 **(E)** and DAPI **(F)**. The number of cells strongly differed between the two measurements (). Thus, the relative frequencies were used for comparison, instead of absolute frequencies. To remove outliers from the plot and increase readability, only datapoints in the 0.95 quantile were plotted. As the two experiments were performed independently, but under the same conditions, all distributions except Rex1/Sox2 were expected to be very similar. Except some minor deviances, the distributions seem very comparable. This implication was affirmed by the calculated correlations between the factors in both experiments (see Table 2 and Table 3), which were very similar, too. Additionally, the similarity of the distributions was checked statistically, as described in Table 1.*

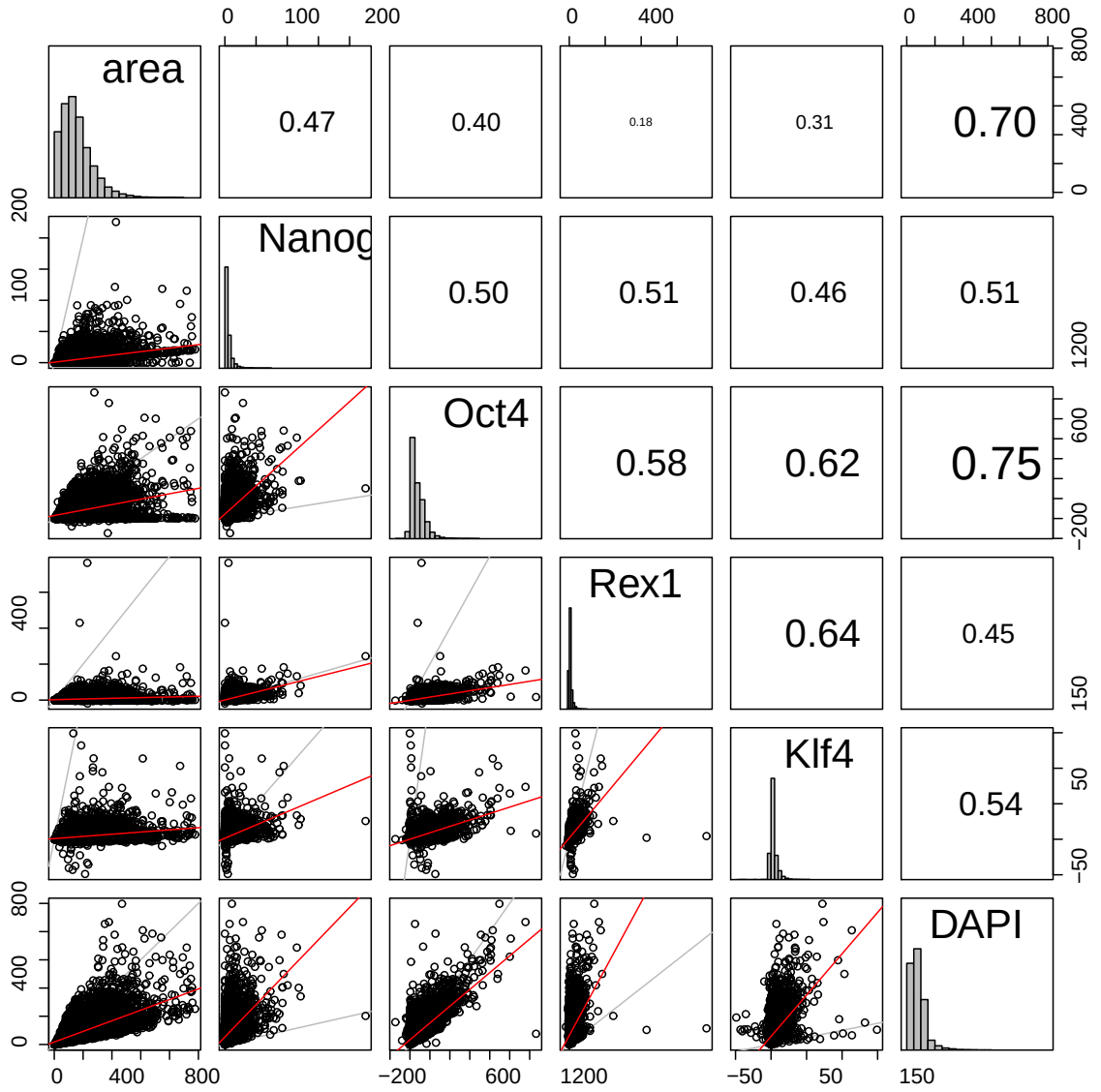

Figure 27: *Cell area, Nanog, Oct4, Rex1, Klf4 and DAPI absolute intensities of the first movie* ***(I)*** *were scatterplotted pairwise in the lower left panels. Red lines present the linear regression of the data displayed in the panel, whereas gray line shows the diagonal line between the two axis. Diagonal panels show the parameter's name, the respective row and column is associated with, as well as a histogram of the datapoints, which are used for plots in the respective column. Panels in the upper right show the Pearson correlations, with the textwidth scaled in respect to the certain values. As expected, area and DAPI correlate very well, whereas the high correlation between Oct4 and DAPI was not explainable, regarding the low correlation between area and Oct4 particularly. Most factors show high pairwise correlations, agreeing with previous findings, which proposed an underlying auto regulatory network, incorporating the pluripotency transcription factors as hubs. [56, 32, 27]*

revealed high Pearson correlations between most factors in general and between the cell area and DAPI ($r = 0.70$), as well as between Oct4 and DAPI ($r = 0.75$) in special. As DAPI strongly binds to A-T rich regions of DNA, its absolute intensity grows with the amount of DNA in cells. As, in turn, the amount of DNA grows with advancing cell cycle as well as the area of the cell, DAPI is expected to show high correlation with a cell's area. Very high correlation of Oct4 with DAPI also indicates cell cycle dependency of Oct4, whereas the correlation $r = 0.4$ between Oct4 and cell area, negates this effect. The high correlations between all of the pluripotency factors emphasise the assumption of a close meshed auto-regulatory network between these transcription factors, as previously described [56, 32, 27].

The overview of movie ***(II)*** (see Figure 28), revealed similar properties, regarding the area and absolute intensities of pluripotency associated transcription factors. Area and DAPI are highly correlated ($r = 0.79$) too, as expected. The correlation between Oct4 and DAPI was even higher ($r = 0.79$) than in the first movie, but still the correlation between Oct4 and area remains low ($r = 0.43$) and puts a possible explanation by cell cycle dependency of Oct4 in question. Also Oct4-Klf4 correlation ($r = 0.79$) is higher than in movie ***(I)*** ($r = 0.62$), as well as Klf4-DAPI (***(I)***: $r = 0.54$, ***(II)***: $r = 0.68$). These differences might have been arisen from slightly different experimental conditions or even the use of the different dyes for Rex1 and Sox2, as the wavelengths of these dyes might slightly overlap with those of dyes for other transcription factors. Rex1 and Sox2 show differences, too: Rex1 correlates better with Nanog ($r = 0.51$) and Klf4 ($r = 0.64$) than Sox2 (Nanog: $r = 0.33$, Klf4: $r = 0.51$), since Rex1 is more associated with Nanog. Shi et al. [48] published, that the Rex1 promoter is located directly downstream of Nanog, and thereby influenced very much.

The presented correlations were similar, in comparison to previous findings. For example a correlation of $r = 0.51$ between Nanog and Oct4 was published by Descalzo et al. [13], and $r = 0.59$ by Thomson et al. [56]. Also the close upstream regulation of Nanog by Klf4, found by Chan et al. [7] and Zhang et al. [64], can be confirmed by our measurements (Nanog-Klf4 ***(I)***: $r = 0.46$ and ***(II)***: $r = 0.48$).

As the correlations could have influenced each other, due to transitive regulations between three or more factors, we decided to further investigate the partial correlations of the pluripotency transcription factors. Additionally slight overlaps in fluorescence wavelengths and other experimental conditions could have led to depending correlations. By calculating partial correlations, linear dependencies are known to be eliminated between different sets of the data [29]. The partial correlation between two independent sets of data (X and Y), influenced by a third one (Z), which is also called the first order partial correlation, can be calculated by

$$pr_{XY,Z} = \frac{r_{XY} - r_{XZ}r_{YZ}}{\sqrt{(1 - r_{XZ}^2)(1 - r_{YZ}^2)}} \tag{4.3.1}$$

, where r_{XY} is the correlation between the sets X and Y. Pictorially, linear regressions are calculated between X and Z as well as Y and Z. The respective residuals of X and Y are correlated against each other in turn, which leads to the partial correlation between X and Y. The final partial correlation between two sets regarding all other sets can recursively becalculated by the second and higher correlations subsequently, until all other variables are regarded by the calculation. The formula for the higher order partial correlations is basically the same as the formula for the first order partial correlations.

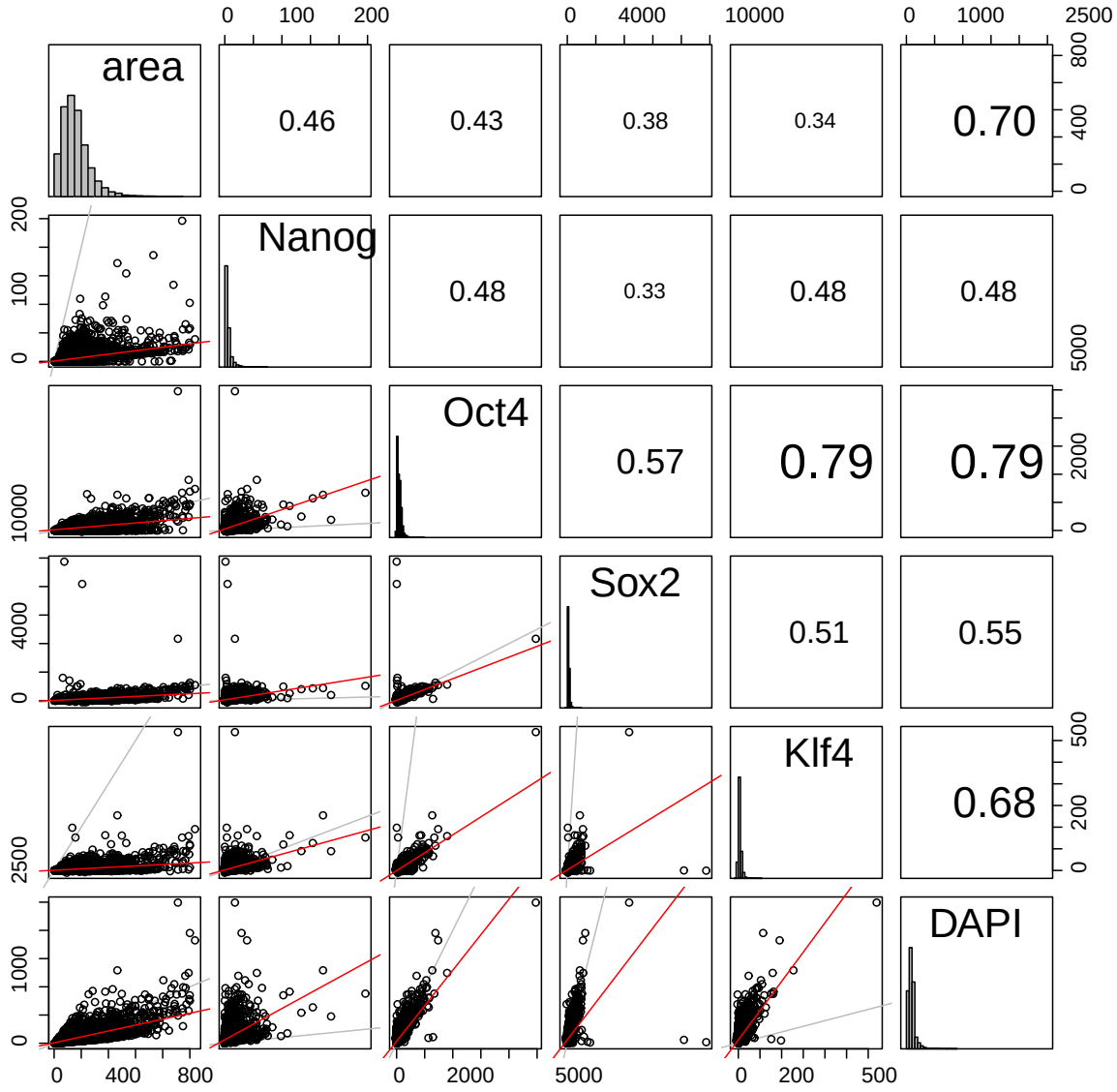

Figure 28: *The lower left panels show pairwise scatterplots of cell area, Nanog, Oct4, Rex1, Klf4 and DAPI of the second movie* ***(II)****. The red lines visualise the linear regressions of the scattered points, whereas the grey line marks the diagonal between the two axis. Calculated Pearson correlations are displayed in the upper right panels, scaled to their value. In between, the diagonal panels display the name of the data sets used in the respective rows and columns, as well as histograms of the data, used for scatterplots in the respective column. DAPI and area correlate very well ($r = 0.70$), as DAPI is a cell cycle marker, showing increasing absolute intensities with ongoing cell cycle and size. Oppositely, the even higher correlation $r = 0.79$ of DAPI with Oct4 lacks this possibility of explanation, as Oct4 does not correlate with cell area that high ($r = 0.40$). Over all, most of the factors correlate high enough to support the assumption of an auto-regulatory network of pluripotency transcription factors, incorporating the previously named as hubs. As expected, area and DAPI correlate very well, whereas the high correlation between Oct4 and DAPI was not explainable, regarding the low correlation between area and Oct4 particularly. Most factors show high pairwise correlations, agreeing with previous findings, which proposed an underlying auto regulatory network, incorporating the pluripotency transcription factors as hubs. [56, 32, 27]*

Table 2: *Pairwise Pearson correlations between the absolute intensities of cell area, Nanog, Oct4, Rex1, Klf4 and DAPI, which were quantified in movie **(I)**, are shown in the upper right. The derived partial correlations are printed in the lower left. Linear dependencies, that influence correlations between the groups are eliminated by calculating partial correlations [29]. The two correlation types are also depicted in Figure 29.*

	area	Nanog	Oct4	Rex1	Klf4	DAPI
area		0.47	0.40	0.18	0.31	0.70
Nanog	0.29		0.50	0.51	0.46	0.51
Oct4	-0.23	0.11		0.58	0.62	0.75
Rex1	-0.20	0.30	0.18		0.64	0.45
Klf4	0.04	0.05	0.23	0.40		0.54
DAPI	0.63	-0.01	0.59	0.06	0.06	

Exemplary, the second order partial correlation between the two sets X and Y, which are controlled by Z and W is

$$pr_{XY,ZW} = \frac{r_{XY,W} - r_{XZ,W} r_{YZ,W}}{\sqrt{(1 - r^2_{XZ,W})(1 - r^2_{YZ,W})}} \tag{4.3.2}$$

The calculated partial correlations between the measured single cell intensities are presented in detail in Table 2 ***(I)*** and Table 3 ***(II)*** and visualised in Figure 29 ***(I)*** and Figure 30 ***(II)*** respectively. Calculation of partial correlations also enables estimation of the p-value, which is associated to a specific correlation. Thereby we filtered for significant correlations, using a Bonferroni corrected significance level of 0.05. Insignificant correlations were disregarded and therefore removed from the graphs. Notably, the correlation and partial correlation graphs of experiment ***(II)*** are similar to those, presented by Filipczyk et al. [17], who used the same images, but performed manual correction and segmentation.

Table 2 shows as well pairwise Pearson correlations (r) as derived partial correlations (pr) between the transcription factors measured in movie ***(I)***. Obviously the values of the partial correlations emphasise some correlations whereas others get low, negative or even insignificant. Of course DAPI-area remains nearly as high as before ($r = 0.70$, $pr = 0.63$), but also Oct4-DAPI, though it decreased, remains high and significant($r = 0.75$, $pr = 0.59$). Further investigations of cell cycle dependencies of Oct4 should reveal the background of this correlation. Also Rex1-Nanog and Rex1-Klf4 remain notably high. Figure 29 depicts these values, and exhibits outstanding correlations more clearly. The correlations between Oct4, Nanog and Klf4 decreased very much. Especially Nanog-Klf4 shows great differences ($r = 0.46$, $pr = 0.05$). In return Rex1 seems to compensate this correlation, as correlation with Nanog ($pr = 0.30$) and Klf4 ($pr = 0.40$) shows much lower decrease. In this point, our results are not fully consistent with previous findings, for example of Chan et al. [7] or Zhang et al. [64], which proposed a direct correlation between Klf4 and Nanog. We, in contrary would assume a regulatory system between Nanog, Klf4 and Rex1, whereas Rex1 transmits regulatory influences between the other two factors. This suggestion has to be confirmed by further investigations of those correlations, to be validated.

Partial correlations of movie ***(II)*** (Figure 30) showed similar properties, as those of movie ***(I)***. Table 3 contrasts Pearson correlations (upper right part) with the partial correlations

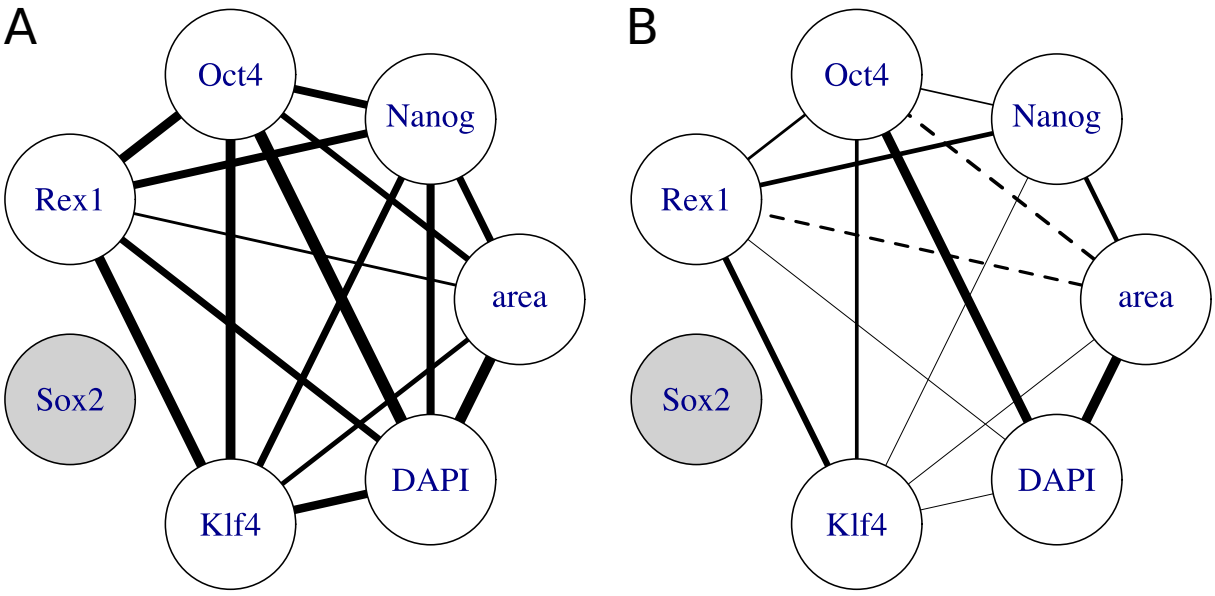

Figure 29: *Table 2, including as well Pearson correlations* ***(A)*** *as partial correlations* ***(B)*** *is visualised. Solid lines represent positive, dashed lines negative correlations. The width of the connections represents the strength of correlation, meaning, the thicker the connection is, the higher is the absolute value of the correlation. In addition to the partial correlations, also the associated p-values were estimated and insignificant correlations removed from the graph. Therefore, a Bonferroni corrected significance level of 0.05 was used.*

Table 3: *Upper right part shows Pearson correlations of absolute intensities of Nanog, Oct4, Sox2, Klf4 and DAPI, quantified in movie* ***(II)****. Partial correlations, calculated from Pearson correlations, are shown in the lower left part and also visualised in Figure 30.*

	area	Nanog	Oct4	Sox2	Klf4	DAPI
area		0.46	0.43	0.38	0.34	0.70
Nanog	0.29		0.48	0.33	0.48	0.48
Oct4	-0.18	0.11		0.57	0.79	0.79
Sox2	0.08	0.01	0.19		0.51	0.55
Klf4	-0.21	0.21	0.45	0.10		0.68
DAPI	0.63	-0.06	0.49	0.08	0.22	

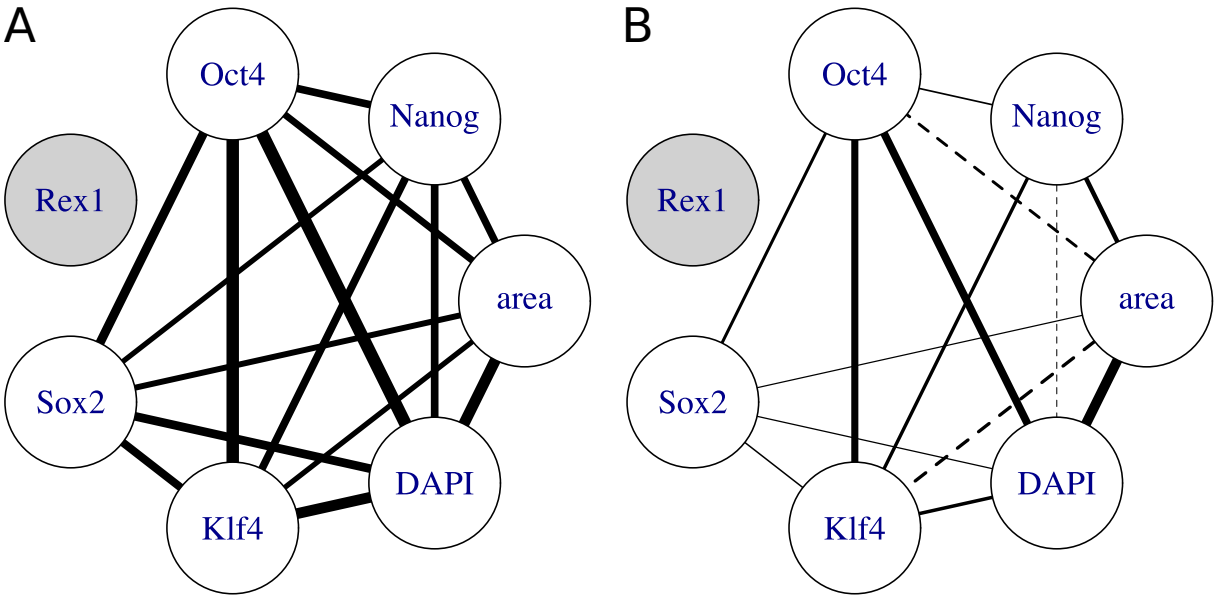

Figure 30: *The graph visualises the correlations, listed in Table 3.* ***(A)*** *shows the Pearson correlations,* ***(B)*** *the derived partial correlations. Line width represents the absolute value of the correlation. Solid lines were plotted for positive correlations, whereas, negative ones are visualised by dashed lines. Missing lines were removed, due to insignificance, regarding the estimated p-value from the partial correlations. Therefore, a Bonferroni corrected significance level of 0.05 was used.*

(lower left). All correlations were decreased, due to removal of linear dependencies between the correlations. The differences between Pearson correlations ***(A)*** and partial correlations ***(B)*** were pointed out in Figure 30. Obviously most correlations decreased clearly or were filtered as insignificant. Area and DAPI remained highly correlated, as expected, but so did Oct4-DAPI, what could not be explained in scope of this work. Interestingly Sox2-Nanog correlation was removed, due to insignificance, contradicting the findings of Rodda et al. [42], who proposed a direct regulation of Nanog by Sox2 and Oct4. Also notably, Oct4 and Klf4 still are highly correlated, which indeed was reported previously [56], but neither detected that clearly, nor investigated in detail to our knowledge.

Notably, correlation between Nanog and Klf4 ($pr = 0.21$) remains higher in partial correlations in movie ***(II)*** than in the dataset using Rex1 ***(II)*** ($pr = 0.05$). We inferred from the high correlations between Rex1 and Klf4 and Rex1 and Nanog in movie ***(I)***, that there existed linear dependencies between these correlations and Klf4-Nanog, which could have been excluded in movie ***(I)***, but not in movie ***(II)***, due to missing measurements of Rex1 intensities.

Correlation between Oct4 and Klf4, also showed a notably high value in movie ***(II)***, and lower but still outstanding value in movie ***(I)***. As -disregarding slight experimental differences- Rex1 and Sox2 are the only difference between the two experiments and their resulting movies, Sox2 was suspected to to be responsible for this drastic difference between the two calculated partial correlations between Oct4 and Klf4 in movie ***(I)*** and ***(II)***.

5 Summary and Outlook

After developing a tool for single cell quantification in microscopy images (*SCQMI*), applying it on ESC fluorescence images to gather snapshot data of certain transcription factors, which are associated with ESC pluripotency and differentiation, and interpreting those data, we here sum up our efforts. In Section 5.1 we briefly sum up our development efforts and the improvements of *SCQMI* in comparison to the prototyped *sQTFy* and other tools and discuss the further development of *SCQMI*. In Section 5.2 we shortly present the results of applying *SCQMI* on ESC fluorescence imaging data and propose further experiments and investigations, based on our findings.

5.1 *SCQMI* - Single Cell Quantification in Microscopy Images

Regarding the application of *SCQMI*, which was developed in scope of this work, we succeeded in complying with the requirements, described in Section 1.4. We created an intuitive program capable of quantifying large-scale imaging experiments, like time lapse fluorescence microscopy, and all precedent steps, like background calculation, correction and segmentation. One main feature of the single cell quantification tool is the underlying data-structure, which enables storage and processing of large-scale experiments containing multiple dimensions and thousands of images. Another innovation is the possibility to create quantification workflows, using a sophisticated graphical user interface and real-time preview of all intermediate results, what makes workflow generation much easier.

As with *sQTFy*, a prototype, written in *MATLAB*, for a single cell quantification tool was already provided, we took advantage of promising parts, like the core workflow, consisting of the background calculation, correction, segmentation and quantification. Other features, we kept in mind, were the usage of *MSER* as advanced segmentation algorithm, the folder structure of TTT experiments for files on hard drive, and the automatic or semi-automatic kind of processing the images. We further improved the graphical user interface and avoided too much pop-ups, which were criticised by users. The workflow concept was established, to cover more use cases and give users the possibility to adapt the image processing to their needs.

Although the current version of *SCQMI* already provides a lot of features, there is still much to improve. For example, more functions have to be implemented as *ImageJ/Fiji* plugins. The still missing background calculation method *Fluorescence Image Normalisation* [46, 47] has to be wrapped as plugin, for example. Other algorithms, that might improve the microscopy images and, therefore, subsequent processing steps like background calculation and segmentation have to be included, too. A contrast enhancement algorithm, like *contrast limited adaptive histogram equalisation* (*CLAHE*) [43, 65], is an exemplary possible candidate for add-ons.

Furthermore, the `Import` part of the tool can indeed use the TTT [37] folder hierarchy to import structured data, but currently, a user has to define the structuring elements of an exemplary image path on his own, as well as the ranges that have to be imported. The graphical user interface can be even more supportive, by detecting a TTT's experiments dimensions and sizes on its own. Also the calculations for real-time preview have to be performed in parallel to the user input, to facilitate even smoother usage.

Last but not least, parts of the software, that simply were out of scope of this work, have to be implemented in the future. As currently, the workflow can only be processed on images sequentially, locally, and in automatic mode, the missing execution modes have to be completed. These are the semi-automatic execution, execution in parallel and execution on a grid engine.

The extensions previously named, can be implemented and added to *SCQMI* easily, as therefore, the extensions were kept in mind and interfaces already prepared during development. Hence the current version of *SCQMI* as well as its future versions, incorporating those extensions, can be used to quantify large-scale microscopy imaging data, as presented in Section 4. A complete list of necessary and realisable extensions to the current version of *SCQMI* is presented in Appendix A. Nevertheless, we discourage any development, that goes beyond those already prepared extensions, as *ImageJ/Fiji* revealed some major drawbacks during development. For example, *ImageJ/Fiji* does not provide any interface for its *Macro Recorder*. As the workflow creation, based on recording of user actions, is one of the main features of *SCQMI*, we alternatively had to query user actions from *ImageJ/Fiji*'s frontend, using a `Listener` class on the macro recordings list of *ImageJ/Fiji*. Another exemplary drawback is the handling of images with different scales, as indeed the value ranges are extended or reduced, but the values simply cut, instead of being rescaled properly. To handle this source of errors, we implemented an additional plugin, that takes care of rescaling the values. As we had to handle those and other difficulties during development, the software grew partly unstructured, what makes further extension challenging. Furthermore, *ImageJ/Fiji* and its underlying datastructure, called *imglib*, are currently redesigned and will be released likely in summer 2014 [11] as *ImageJ2* and *imglib2*. Therefore, we advise to extend the current version of *SCQMI* only as far as needed and to use *ImageJ2* to redesign *SCQMI*, when released. The current roadmap of *ImageJ2* promises a lot of improvements, that will help to implement *SCQMI* more structured. Nevertheless, we propose to check other available tools and frameworks in the meantime, as there exist many more with also promising features, like *FARSight* [16].

5.2 Correlations between ESC pluripotency associated transcription factors

By applying *SCQMI* successfully to quantify two different movies of cultured ESCs, showing some single cell intensities of transcription factors, associated with the regulation of pluripotency and differentiation, we were able to gather large-scale quantification data. Analysis of this data, in comparison to previous findings, revealed some already known correlations between the factors, as well as some still undescribed correlations. Before the specific transcription factors can be investigated in more detail, a few experiments, similar to the one, described in this work, have to be performed. Thereby, the correctness and consistency of our findings can be validated.

Although the similarity of the distributions, derived from the quantifications, was shown in Section 4.3, the standard non-parametric statistical tests like the Kolmogorov-Smirnov or Wilcoxon signed-rank test rejected the null hypothesis, stating the pairwise equality of the distributions. The reason for the detected difference could not be explained in scope of this work and has to be investigated in detail, as all interpretations of the comparisons between the two experiments rely on their comparability.

During data analysis, we found new correlations, which were not yet described in literature (high Oct4-DAPI correlation and reducing Klf4-Nanog correlation in the dataset, including Rex1 ***(I)***). These have to be investigated in detail, meaning further statistical investigations of the existing datasets, as well as further experiments, hopefully unveiling the underlying regulatory system of the correlations. Especially the significant reduction of the Klf4-Nanog correlation, by including Rex1 into calculation of partial correlations is interesting and points to indirect regulation between Nanog and Klf4, using Rex1 as intermediate regulator.

In addition, experiments, simultaneously containing intensities of Sox2 and Rex1 have to be performed, to capture effects between these two factors, too. In best case, these experiments investigate all regarded transcription factors all in once, for example by Quantum dots. *Quantum dots* promise much better fluorescence images, by providing greater resolution as well as tighter wavelength spectra, enabling more transcription factors being investigated in one experiment with simultaneously clearer separable fluorescence signals [59].

To enhance analysis and interpretation, the cells have to be evaluated individually or colony-wise, but not united to populations. Also, the time resolution has to be taken into account. Filipczyk et al. [17] for example, identified colonies, defined as descendants of the same cell, that all show an intensity of Nanog of a certain level. This level can be either colony-wise low, or low and high, named mosaic. Filipczyk et al. [17] described a clear difference between the correlations of the measured pluripotency transcription factors, in the two data subsets. As the time resolved data of Nanog intensities already exists, we propose further analysis, based on a decision tree, to reveal the certain levels and correlations, that lead to the differentiation of those different colonies. Therefore, the lineage tree can be mapped on a decision tree of certain Nanog levels. By transcloning a second marker into another pluripotency transcription factor gene (like Klf4), even the correlations between these two transcription factors can be taken as a basis for the decision tree.

Concluding, we developed and presented a tool for single cell quantification in microscopy images, providing all needed features in an easy and intuitive graphical user interface. As the backend components were developed under consideration of very large-scale applications, the tool is capable of loading and processing a whole experiment's data. The performed workflow is not predefined by the software, but can be easily defined by a user. Thus *SCQMI* can be used on a broad range of different images, like fluorescence or even bright-field images. We exemplary used *SCQMI* on two imaging experiments, resulting in quantifications of cell-wise intensities of transcription factors, associated with pluripotency. These quantifications were analysed, compared to already published findings, and further investigations proposed, on base of findings, not yet described in literature. With the development and applications of this work, we presented promising possibilities, which will help understanding the principles behind ESC pluripotency and differentiation and further biological questions, by supporting large-scale quantification of single cells in microscopy images.

A Future Extensions of *SCQMI*

- Automatic detection of TTT experiments including number and type of dimensions as well as their sizes (see Figure 4).
- Implementation of a quick preview possibility of selected images in the `VirtualData` window.
- Implementation of the *Fluorescence Image Normalization* proposed by [47] as background calculation.
- Implementation of the *Subtract and Gain* method for correction. The standardised location of the gain image has to be fixed in the code and associated with the right images for division.
- Improvement of the segmentation method. As currently *MSER* is the de-facto standard of *SCQMI*, it can be extended like proposed by Hilsenbeck O., who tried to incorporate morphological features into *MSER*.
- Supply of some compiled versions of *MSER* for *Widnows* and *Linux*, as *MSER* uses the Java Native Library, and has to be compiled on every system, before being used.
- Extension of the quantification step in the workflow generator by automatic statistic analysis and display of certain measured features, for example by histograms.
- Implementation of serialisation and de-serialisation of the workflow object, to save and load workflows.
- Development of a function to check, whether a certain workflow can be calculated automatically, or it requires semi-automatic processing, due to obligatory user input, before bulk execution.
- Implementation of bulk execution on a grid as well as the associated graphical user interface to adequately monitor the proceedings.
- Implementation of multi-threaded bulk execution on a grid as well as locally. Possible errors have to be caught and handled appropriately.
- Implementation of user friendly error handling, as errors are still just reported to the log window but handled not further.
- Development of an install script or makefile, as installation of certain parts of *SCQMI*, especially those that use the Java Native Library, might be challenging for some users.
- Addition of further tooltips and creation of a useful documentation.

References

[1] **G. A. Archanjo**. "Marvin Image processing Framework)". `http://marvinproject.sourceforge.net/en/index.html`. Sept. 2013.

[2] **M. A. Arias** and **J. M. Brickman**. "Gene expression heterogeneities in embryonic stem cell populations: origin and function." In: *Current opinion in cell biology* 23.6 (Dec. 2011), pp. 650–6.

[3] **G. Bradski**. "The OpenCV Library". In: *Dr. Dobb's Journal of Software Tools* (2000).

[4] **T. Burdon**, **A. Smith**, and **P. Savatier**. "Signalling, cell cycle and pluripotency in embryonic stem cells." In: *Trends in cell biology* 12.9 (Sept. 2002), pp. 432–8.

[5] **A. E. Carpenter** et al. "CellProfiler: image analysis software for identifying and quantifying cell phenotypes." In: *Genome biology* 7.10 (Jan. 2006), R100.

[6] **I. Chambers** et al. "Nanog safeguards pluripotency and mediates germline development." In: *Nature* 450.7173 (Dec. 2007), pp. 1230–4.

[7] **K. K.-K. Chan** et al. "KLF4 and PBX1 directly regulate NANOG expression in human embryonic stem cells." In: *Stem cells (Dayton, Ohio)* 27.9 (Sept. 2009), pp. 2114–25.

[8] **C. Chen** et al. "A general system for automatic biomedical image segmentation using intensity neighborhoods." In: *International journal of biomedical imaging* 2011.8 (Jan. 2011), p. 606857.

[9] **L. P. Coelho**, **A. Shariff**, and **R. F. Murphy**. "Nuclear segmentation in microscope cell Images: A hand-segmented dataset and comparison of algorithms." In: *Proceedings / IEEE International Symposium on Biomedical Imaging: from nano to macro. IEEE International Symposium on Biomedical Imaging* 5193098 (Jan. 2009), pp. 518–521.

[10] **V. C. Coffman** and **J.-Q. Wu**. "Counting protein molecules using quantitative fluorescence microscopy." In: *Trends in biochemical sciences* 37.11 (Nov. 2012), pp. 499–506.

[11] **I. community**. "ImageJ2 Development". `http://trac.imagej.net/roadmap`. Sept. 2013.

[12] **D. L. Coutu** and **T. Schroeder**. "Probing cellular processes by long-term live imaging - historic problems and current solutions." In: *Journal of cell science* 126.August (Aug. 2013), pp. 3805–3815.

[13] **S. M. Descalzo** et al. "Correlations between the levels of Oct4 and Nanog as a signature for naive pluripotency in mouse embryonic stem cells." In: *Stem Cells* 30.12 (2012), pp. 2638–2691.

[14] **G. P. C. Drummen**. "Fluorescent probes and fluorescence (microscopy) techniques–illuminating biological and biomedical research." In: *Molecules (Basel, Switzerland)* 17.12 (Jan. 2012), pp. 14067–14090.

[15] **H. M. Eilken**, **S.-I. Nishikawa**, and **T. Schroeder**. "Continuous single-cell imaging of blood generation from haemogenic endothelium." In: *Nature* 457.7231 (Feb. 2009), pp. 896–900.

[16] "FARSIGHT Project Wiki". `http://www.farsight-toolkit.org/wiki/Main_Page`. Sept. 2013.

[17] **A. Filipczyk** et al. "Network heterogenity of Pluripotency Transcription Factors in Embryonic Stem Cells".

[18] **M. Grev**. "MigLayout". http://wwww.MigLayout.com. Aug. 2013.
[19] **N. I. of Health**. "ImageJ Processes". http://rsbweb.nih.gov/ij/docs/menus/process.html. Aug. 2013.
[20] **Z. Hensel** and **J. Xiao**. "Single-molecule methods for studying gene regulation in vivo." In: *Pflügers Archiv : European journal of physiology* 465.3 (Mar. 2013), pp. 383–95.
[21] **O. Hilsenbeck**. "Automated construction of cell lineage trees from time-lapse microscopy data". Bachelor Thesis. Ludwig-Maximilian Universität München, 2011.
[22] **A. M. Jones** et al. "In vivo biochemistry: applications for small molecule biosensors in plant biology." In: *Current opinion in plant biology* 16.3 (June 2013), pp. 389–395.
[23] **M. Jones**. "Stem Cell Development". http://en.wikipedia.org/wiki/Embryonic_stem_cell. Sept. 2013.
[24] **P. Kainth** and **B. Andrews**. "Quantitative cell array screening to identify regulators of gene expression." In: *Briefings in functional genomics* 9.1 (Jan. 2010), pp. 13–23.
[25] **D. Kamiyama** and **B. Huang**. "Development in the STORM." In: *Developmental cell* 23.6 (Dec. 2012), pp. 1103–1110.
[26] **G. M. Keller**. "In vitro differentiation of embryonic stem cells." In: *Current opinion in cell biology* 7.6 (Dec. 1995), pp. 862–9.
[27] **J. Kim** et al. "An extended transcriptional network for pluripotency of embryonic stem cells." In: *Cell* 132.6 (Mar. 2008), pp. 1049–1061.
[28] **S. Klein**. "elastix: a toolbox for rigid and nonrigid registration of images". http://elastix.isi.uu.nl. Aug. 2013.
[29] **S. L. Lauritzen**. "Graphical Models(Google eBook)". Oxford University Press, 1996, p. 308.
[30] **W. A. Lim**, **C. M. Lee**, and **C. Tang**. "Design principles of regulatory networks: searching for the molecular algorithms of the cell." In: *Molecular cell* 49.2 (Jan. 2013), pp. 202–12.
[31] **S. van de Linde** et al. "Investigating cellular structures at the nanoscale with organic fluorophores." In: *Chemistry & biology* 20.1 (Jan. 2013), pp. 8–18.
[32] **B. D. MacArthur** et al. "Nanog-dependent feedback loops regulate murine embryonic stem cell heterogeneity." In: *Nature cell biology* 14.11 (Nov. 2012), pp. 1139–47.
[33] **G. R. Martin**. "Isolation of a pluripotent cell line from early mouse embryos cultured in medium conditioned by teratocarcinoma stem cells". In: *Developmental Biology* 78.12 (1981), pp. 7634–7638.
[34] **J. Matas** et al. "Robust Wide Baseline Stereo from Maximally Stable Extremal Regions". In: *Procedings of the British Machine Vision Conference 2002* 36 (2002), pp. 1–10.
[35] **MathWorks**. "Mathwork's MATLAB". http://www.mathworks.de/products/matlab/. Aug. 2013.
[36] **A. Miyawaki**. "Fluorescence imaging in the last two decades." In: *Journal of electron microscopy* 62.1 (Jan. 2013), pp. 63–8.
[37] **H. Z. Mnchen**. "Timm's Tracking Tool". http://www.helmholtz-muenchen.de/scd/service/scientific-services/software-downlo
Aug. 2013.

[38] **T. Nagai** et al. "A variant of yellow fluorescent protein with fast and efficient maturation for cell-biological applications." In: *Nature biotechnology* 20.1 (Jan. 2002), pp. 87–90.

[39] **N. Otsu**. "A threshold selection method from gray-level histograms." In: *IEEE Transactions on Systems, Man, and Cybernetics* 9.1 (1979), pp. 62–66.

[40] **U. U. o. C. Renato Archer Information Technology Center**. "Phyton Morphology Toolbox". `http://adessowiki.fee.unicamp.br/adesso/wiki/ia870/view/`. Sept. 2013.

[41] **M. A. Rieger** et al. "Hematopoietic cytokines can instruct lineage choice." In: *Science (New York, N.Y.)* 325.5937 (July 2009), pp. 217–8.

[42] **D. J. Rodda** et al. "Transcriptional regulation of nanog by OCT4 and SOX2." In: *The Journal of biological chemistry* 280.26 (July 2005), pp. 24731–7.

[43] **S. Saalfeld**. "CLAHE (Contrast Limited Adaptive Histogram Equalization)". `http://rsbweb.nih.gov/ij/plugins/clahe/index.html`. Sept. 2013.

[44] **A. K. Saxena**, **D. Singh**, and **J. Gupta**. "Role of stem cell research in therapeutic purpose–a hope for new horizon in medical biotechnology." In: *Journal of experimental therapeutics & oncology* 8.3 (Jan. 2010), pp. 223–33.

[45] **J. Schindelin** et al. "Fiji: an open-source platform for biological-image analysis." In: *Nature methods* 9.7 (July 2012), pp. 676–82.

[46] **M. Schwarzfischer** et al. "Efficient and reliable long-term single-cell tracking and quantification of cellular and molecular behaviour in time-lapse microscopy".

[47] **M. Schwarzfischer** et al. "Efficient fluorescence image normalization for time lapse movies." In: *International Conference on Systems Biology*. Institute of Bioinformatics and Systems Biology of the Helmholtz Zentrum Munich, Germany. Heidelberg, 2011.

[48] **W. Shi** et al. "Regulation of the pluripotency marker Rex-1 by Nanog and Sox2." In: *The Journal of biological chemistry* 281.33 (Aug. 2006), pp. 23319–25.

[49] **S. Siegel**. "Nonparametric statistics for the behavioural sciences". McGraw-Hill, 1956, p. 312.

[50] **N. V. Smirnov**. "Tables for estimating the godness of fit of empirical distributions." In: *Annals of Mathematical Statistics* 19 (1948), p. 279.

[51] **M. A. Stephens**. "EDF Statistics for Goodness of Fit and Some Comparisons". In: *Journal of the American Statistical Association* 69.347 (1974), pp. 730–737.

[52] **S. R. Sternberg**. "Biomedical Image Processing". English. In: *Computer* 16.1 (Jan. 1983), pp. 22–34.

[53] **U. de Strasbourg**. "medipy - A medical image processing framework)". `http://code.google.com/p/medipy/`. Sept. 2013.

[54] **F. A. A. Talbot**. "Practical cinematography and its applications". W. Heinemann, 1913.

[55] **E Tholouli** et al. "Quantum dots light up pathology". In: *Journal of Pathology* 216.July (2008), pp. 275–285.

[56] **M. Thomson** et al. "Pluripotency factors in embryonic stem cells regulate differentiation into germ layers." In: *Cell* 145.6 (June 2011), pp. 875–89.

[57] **E. R. Tkaczyk** and **A. H. Tkaczyk**. "Multiphoton flow cytometry strategies and applications." In: *Cytometry. Part A : the journal of the International Society for Analytical Cytology* 79.10 (Oct. 2011), pp. 775–88.

[58] **R. Usamentiaga** et al. "Automatic detection of impact damage in carbon fiber composites using active thermography." In: *Infrared Physics & Technology* 58 (May 2013), pp. 36–46.

[59] **M. a. Walling**, **J. a. Novak**, and **J. R. E. Shepard**. "Quantum dots for live cell and in vivo imaging." In: *International journal of molecular sciences* 10.2 (Feb. 2009), pp. 441–91.

[60] **F. Wilcoxon**. "Individual Comparisons by Ranking Methods". In: *Biometrics Bulletin* 1.6 (1945), pp. 80–83.

[61] **G. Wu** et al. "Establishment of totipotency does not depend onOct4A". In: *Nature Cell Biology* 15.9 (Aug. 2013), pp. 1–11.

[62] **J. Wu** et al. "ChIP-chip comes of age for genome-wide functional analysis." In: *Cancer research* 66.14 (July 2006), pp. 6899–902.

[63] **Q.-L. Ying** et al. "The ground state of embryonic stem cell self-renewal." In: *Nature* 453.7194 (May 2008), pp. 519–23.

[64] **P. Zhang** et al. "Kruppel-like factor 4 (Klf4) prevents embryonic stem (ES) cell differentiation by regulating Nanog gene expression." In: *The Journal of biological chemistry* 285.12 (Mar. 2010), pp. 9180–9.

[65] **K. Zuiderveld**. "Contrast limited adaptive histogram equalization". In: *Graphic Gems IV*. Academic Press Professional, Inc., Aug. 1994, pp. 474–485.