LifeXplorer

Improves Image Acquisition and Analysis in Systems Biology

Dissertation

for the degree of

Doctor of Natural Sciences (Dr. rer. nat.)

submitted to the

Combined Faculties for the Natural Sciences and Mathematics

of the Ruperto-Carola University of Heidelberg, Germany

submitted by

Jonas Carl Moßler, Dipl.-Inf.(TI)

born in Achim

Heidelberg, 2013

Advisors: Prof. Dr. Roland Eils

Prof. Dr. Peter Fischer

Oral examination:

For science & my family

Abstract

Improvements in the capacity to perform and analyze time-resolved experiments are a fundamental challenge in quantitative biology, with automation providing the eventual solution. To that end the author developed an algorithm, termed Automated Result / Cost Optimization (ARCO), as a universally-applicable approach based on an operator-defined experiment result. ARCO increased the information density of high-resolution microscopy experiments while minimizing the associated costs, including the loss of biosensor sensitivity and phototoxicity. Several derivations of the ARCO algorithm provided automatic optimization of microscopy experiments via dynamic re-adjustments of parameters, including XY-positions and light exposure during run time. Furthermore, to integrate high-throughput microscopy with single-cell, high-sampling online data analysis, the author developed LifeXplorer, an adaptable processing and hardware control platform, here applied to the Olympus IX81 ScanR and Nikon Ti-E NIS Elements imaging systems. All applications of the ARCO algorithm were benchmarked against two intracellular events which are well known to be influenced by phototoxic stress; mitochondrial energetics and the reassembly of the Golgi. ARCO optimization significantly increased the information content and accuracy of experimental results, while reducing phototoxicity by several fold. The LifeXplorer platform integrates commonly-used tools (ImageJ, CellProfiler, Matlab, etc.), allowing for adoption and development of new ARCO applications. The application of ARCO to both image acquisition and analysis is an important step towards automation and integration of microscopy and data analysis for the emergence of quantitative biology via computer vision.

Zusammenfassung

Die Möglichkeiten zu erweitern, Lebendzellexperimente optimaler durchführen und auswerten zu können, ist eine fundamentale Herausforderung im Bereich der quantitativen Biologie. Aus diesem Grunde entwickelte der Autor den Automated Result / Cost Optimization (ARCO) Algorithmus, als einen universell einsetzbaren Ansatz, um die Effizienz und Effektivität wissenschaftlicher Experimente automatisch zu erhöhen. Der Ansatz basiert auf der a priori Modellierung des Experimentergebnisses durch den Anwender. ARCO verbesserte den Informationsgehalt von hochauflösenden Mikroskopie-Experimenten, während die damit in Verbindung stehenden Kosten minimiert wurden, einschließlich des Empfindlichkeitsverlusts von Biosensoren sowie der Phototoxizität der Messungen. Abgeleitet vom ARCO Algorithmus konnten mehrere Methoden entwickelt werden, die unterschiedliche Parameter von Mikroskopie Experimenten während der Laufzeit durch dynamische Anpassung optimieren. Dazu gehörten u.a. die Optimierung der XY-Position und der Belichtungszeit. Für die Integration des Algorithmus in die Hochdurchsatzmikroskopie und zur Optimierung von Einzelzellstatistiken bei hoher zeitlicher Auflösung wurde LifeXplorer entwickelt. LifeXplorer ist eine flexible Plattform zur Steuerung von Hardwarekomponenten und Echtzeitdatenverarbeitung, die hier an zwei unterschiedliche Hochdurchsatz Mikroskopie Systeme gekoppelt wurde: Olympus IX81 ScanR und Nikon Ti-E NIS Elements. Alle Anwendungen von ARCO wurden auf Basis zweier intrazellulärer Vorgänge untersucht und bewertet: die mitochondriale Energetik und der Wiederaufbau des Golgi-Apparats. Beide Vorgänge sind bekannt dafür, dass sie durch phototoxischen Stress beeinflusst werden. Die Optimierung mit ARCO konnte den Informationsgehalt und die Genauigkeit der Experimente signifikant erhöhen, während die Phototoxizität stark reduziert werden konnte. Die LifeXplorer Plattform integriert im Bereich der quantitativen Biologie häufig genutzte Standardwerkzeuge wie ImageJ, CellProfiler, Matlab und weitere. LifeXplorer ist damit leicht anpassungsfähig und ermöglicht die Entwicklung neuer ARCO Anwendungen. Die Anwendung von ARCO ist ein wichtiger Schritt in Richtung Automatisierung und Integration von Mikroskopie und Datenanalyse, um die Wirksamkeit quantitativer Biologie mithilfe von Computern zu steigern.

Contents

1 Introduction 7

1.1 Motivation 10

1.2 Research Question and Approach 13

1.3 Contributions 16

1.4 Thesis Outline 17

2 Method: Automated Result / Cost Optimization 18

2.1 Abstract 18

2.2 Introduction 19

2.3 Results 20

2.4 Discussion 27

2.5 Material and Methods 29

2.6 Supplementary Methods 30

2.7 Supplementary Figures 32

3 Implementation: the LifeXplorer platform 39

3.1 Introduction 41

3.2 Basic Technologies for Imaging and Data Processing 45

3.3 Software Architecture 77

4 Applications: of the ARCO algorithm 110

4.1 ASEC (Application Specific Exposure Control) 110

4.2 Autophagy dynamics with ASEC 127

4.3 Mitochondrial energetics with ASEC 133

4.4 Golgi reassembly with ASEC 138

4.5 MBAAS (Model Based Autofocusing Algorithm Selection) 144

4.6 RFT (Re-Focusing Trigger) 172

5 Conclusion and Future work 188

5.1 Conclusion 188

5.2 Future work 190

Acknowledgements 203

Bibliography 208

Chapter 1
Introduction

Managing complexity in quantitative biology is a major challenge. Automating the process of data acquisition and analysis as far as possible is a key factor in this science field. To generate insights into life, extended interdisciplinary knowledge is needed. This knowledge often spreads into a variety of scientific disciplines including biology, microscopy, imaging, image processing, statistics and modeling. Subjective human experiences, however, affect the pipeline of insight generation. The generalizability of results is therefore prone to be experience related. The reproducibility of quantitative scientific results becomes less dependable. Sources of error exist in each work step from the biological setup, over the measurements to the modeling. This often leads to different results. This issue of subjectivity in science has to be addressed and overcome with automation. For automation, mathematical models are necessary. They can reduce the impact of subjective experiences. The author's hypothesis was that intelligent microscopy based on previous knowledge and mathematical models increases the reproducibility and the information density of biological experiments. Five basic technical parameters play a crucial role in live cell imaging. These include the three-dimensional position XYZ, the exposure of light, the magnification, the resolution and the sampling rate. Several focusing algorithms and strategies were evaluated in literature [1-3] in order to find the in-focus plane. To find the relevant information region of interest, algorithms were applied in offline and online processing steps [4]. Controlled Light Exposure Microscopy (CLEM) for laser scanning microscopes showed that the phototoxicity [5, 6] and photobleaching [7-9] effects in imaging can be reduced if background pixels are exposed to a different exposure time as foreground pixels [10, 11]. How the light dose influences the viability of cells was also investigated, although no established standard exists to approximate phototoxic effects quantitatively [6]. Trusting the computer to make decisions and to find relevant phenotypes is an approach increasingly used in quantitative biology [12]. The

combination of online data analysis with the observation process helps to increase the information density and to find relevant information on high dimensional information. By that, integrated automation of data acquisition and analysis reduces complexity for biologists, which was already demonstrated for a machine learning application that identifies a phenotype [4] on run time.

In this thesis, the author presents novel approaches to optimize the parameter space of live cell imaging experiments with a unique algorithm called Automated Result / Cost Optimization (ARCO). From this algorithm, several methods were derived, evaluated and implemented in LifeXplorer. LifeXplorer is a platform that makes it easy to integrate intelligent microscopy strategies. It uses existing tools such as ImageJ [13], Matlab, Mathematica, CellProfiler [14], and others. Using these standards enables us to demonstrate the capacity of self-organized computing networks to determine information density and quality optimization steps, which can be used to increase significantly the outcome of biological experiments. The first method that is run on the LifeXplorer platform is an application specific exposure control (ASEC) that reduces phototoxicity by use of a global light exposure control. Based on a model of the quantity that should be observed in each channel, a classification efficiency is computed and the lowest exposure time possible is configured. The method was applied to mitochondrial energetics which could be observed three- to sixfold longer before becoming apoptotic. Imaging the reassembly of the Golgi apparatus after observing the disassembly was not possible using the operator's guess for a suitable exposure time; however, it was possible using ASEC to mathematically decide which exposure time is necessary. In order to increase information density, LifeXplorer observes as many cells as possible with a region of interest control logic the author called OSAPI (Optimized Sampling Point Identification). It establishes optimized sampling points. The autofocus function is optimized by an automated selection of the best focusing algorithm. The method behind is based on a real-time modeling of the surface in order to get better estimates for the focus plane in the focusing step and to automatically select the best suitable focusing algorithm for the sample. The refocusing is optimized by a refocus trigger (RFT) logic that triggers refocusing, depending on image quality and position. The

sampling rate decreases automatically according to each channel. This decrease depends on cut-off frequency detection (CDF). Implementing different forms of quality control and information densification strategies to automate high-throughput microscopy and data analysis based on ARCO leads, however, to technical challenges. LifeXplorer addresses these challenges, such as dependability and redundancy for memory afflicted, sampling position and channel related data analysis. This study is aimed to demonstrate relevant applications of intelligent microscopy using the ARCO algorithm implemented in LifeXplorer. LifeXplorer was benchmarked against two intracellular events influenced by imaging induced phototoxic stress: mitochondrial energetics and the reassembly of the Golgi. ARCO optimization significantly increased the information content and accuracy of live-cell experimental results, while reducing phototoxicity 3-6 fold. With ARCO the author introduces an overall system optimization for microscopy data acquisition and analysis.

1.1 Motivation

1.1.1 Ethical Impact of Life Sciences

Knowledge is a cornerstone of all conscious actions of human beings. Information and the generation of insights into the laws of life as well as structural system-related knowledge are fundamental for humanity and culture. Ethics, as a practical discipline of philosophy, is engaged with the ethos, the habits and customs of people. Aristotle was the first philosopher to discuss ethos (384 B.C. in Stageira; † 322 B.C. in Chalkis); it had become a central philosophical discipline by the time of Socrates (469 B.C. in Alopeke, Athen; † 399 B.C.) after the so called Socratic shift. The Socratic shift changed the focus of philosophy from the observation of nature towards the reflection of the human. Sophists believed that following traditions and conventions is unreasonable for humans as rational beings. Humans should rather act according to rational models that are created out of reflection and are logically established. Philosophers including Pythagoras (570 B.C; † 510 B.C.) reinvented the natural sciences on the basis of philosophy and its methods. These approaches to thinking merged with existing knowledge systems, i.e. mathematics, becoming fields such as astrology and physics. Using mathematics as a basic tool to model laws and processes, these science fields become quantitative. The development of a system's behavior thus can be predicted over time within a certain environment. The ethical impact of the scientific methodology's ability to enable humans to predict the future is significant. It empowers humans to build and control dependable systems in reflection of Aristotle's understanding of ethics. Armed with this quantitative approach, humans are able to rule the world on the basis of natural laws. This task was already given in the Bible (Genesis 1:28). However sustainable power does not come without responsibility. Otherwise power becomes destructive and what was destroyed cannot be controlled anymore. Western culture gives a feeling that the next step of human evolution,

control over biological systems is both: an indispensable subtask of the human's duty and a challenge for ethics. Quantitative biology enables the formulation of dependable predictions. The robust control of biological systems then emerges into synthetic biology. Bio-engineering can contribute to heal diseases like cancer. It can also help to develop drug treatments and therapies that are more specifically in their influence on the human body with personalized medicine.

1.1.2 Technical Impact of This Work

The automation of life sciences is a consequent step, as the existing knowledge and models of the world are fundamental for the interpretation of scientific results, which can be explained by the philosophy and theory of science [13]. The work described in this thesis is motivated by this idea. Existing knowledge that can be formulated mathematically should be exchangeable in standard formats and integrated into instruments, such as high-throughput microscopes, in order to increase the information density of the measured data. Observing living cells and organisms with fluorescence [14] microscopy is made more difficult by major technical limitations such as optical resolution, imaging throughput, phototoxicity and photobleaching [5]. At the same time, data analysis is limited by available storage and computing resources, data volumes and processing speed. The *main hypothesis driving this work is that integrating previous knowledge, along with questions about the biological system, into the microscope can help to increase the information density of experiments while decreasing the amount of necessary data.* This certainly can be explained by information and system theory. Mathematically, the technical parameter space of the observation needs to be sampled to find the relevant information to answer a specific scientific question. Several computational issues arise with the complexity of the described problem. The larger the range of the parameters are, the more dimensions have to be taken into account [15]. Several strategies were developed to find suitable parameters to solve a given problem, or in other words to describe a system in which its expected behavior is based on a predefined or

automatically generated model. Data mining i.e. is a common strategy for data analysis. It is applicable to find functional relations between dimensions and to extract structural information of data, such as data clusters.

The generation of insights and the creation of robust models of life is a challenging task. It consists of a complex variety of different theoretical and practical questions. Additionally, observing living is challenging. Both the observation and the biological system become defective. Biological systems are sensitive to their environment. The chemical environment, the temperature and the light all play a central role in live cell imaging. Technical parameters, such as the light dose and the sampling rate, therefore can directly influence the biological system under observation. The term phototoxicity is used to describe this phenomenon. Altering the behavior of the biological system by the influence of the measurement is undesirable. Disturbing signals appearing in data acquisition are multiplied in the data analysis workflows. They hold the potential of crucially decreasing the value of experimental results. Today, the bases for choosing the technical parameters for live cell imaging are experience and individual knowledge. Integrating mathematical formulation and questions about the biological system leads to reproducible and generalizable decisions. Models, data analysis, and decision support systems allow optimizing the parameter space. Being shared digitally, the exchange of scientific knowledge can be enhanced. Transforming knowledge and strategies into more generalizable units would lead to more efficient access to knowledge. It would improve how we currently observe and analyze specific biological systems. These units can then be used by instruments in an evolving process to autonomously make or suggest decisions on how to set up, image, and analyze biological systems in a mathematically optimized way.

1.2 Research Question and Approach

The goal of this dissertation is to optimize the efficiency of scientific experiments in the context of high-throughput microscopy for life sciences. This research introduces a new algorithm called Automatic Experiment Efficiency Optimization (ARCO) as a generalized approach to increase the efficiency of scientific experiments. It demonstrates practical and novel applications of intelligent microscopy and data processing, derived from ARCO. To be able to test several applications of ARCO a platform for intelligent microscopy and processing, called LifeXplorer, was built. LifeXplorer was connected with two different hardware platforms. This thesis therefore introduces ARCO and the LifeXplorer platform (hosting ARCO applications) as a case study of a higher level of automating life sciences, with the aim to increase both productivity and reproducibility. Infrastructural challenges were addressed in the LifeXplorer platform, regarding how to efficiently integrate knowledge into imaging workflows of microscopes. Several applications were built as case studies to control the quality of images and to optimize the parameterization by the usage of existing models about the biological system and the measured quantities.

The main hypothesis motivating this research was:

> *"Models of biological systems can be used in automated acquisition and analysis systems to optimize parameters of scientific experiments. Automated systems thus enhance information density and content."*

Figure 1 illustrates the general idea of the ARCO algorithm. This dissertation is motivated and based on the generalized approach of ARCO to optimize the efficiency of scientific experiments by automating parameter configuration decisions.

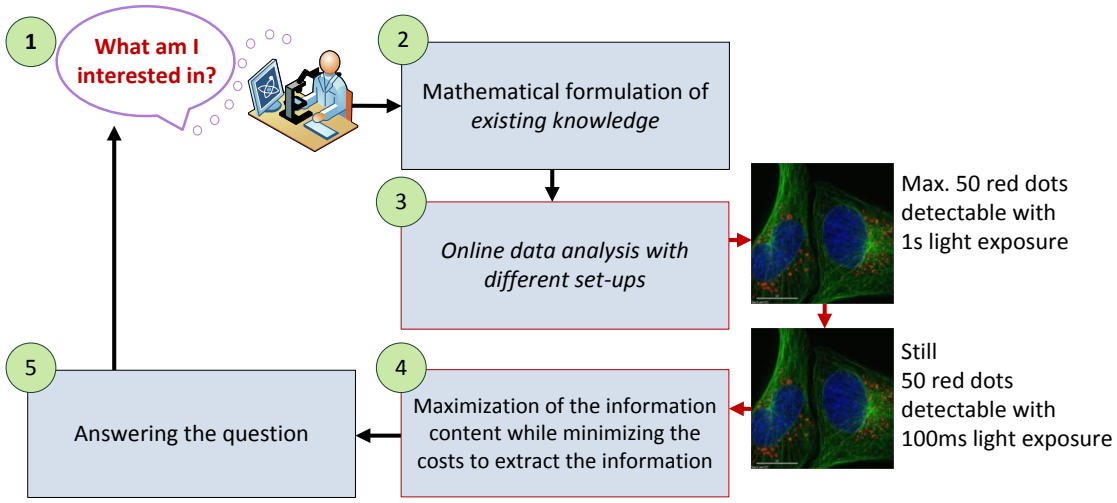


Figure 1 The ARCO algorithm. The ARCO algorithm to optimize the efficiency of one experimental dimension. First the scientific question is defined. Second the question has to be formulated in means of math using relevant existing knowledge to answer the question, i.e. models about the quantities involved in the question. Third, the parameter of one experimental dimension is swept within a defined range, and an online data analysis extracts relevant information. Fourth, the minimal costs are detected to extract the maximal information content necessary to answer the question. Fifth, the experiment is run and additional steps are executed to answer the scientific question.

The experimental set-up and approach for integrating models of existing knowledge about the experiment is illustrated in **Figure 2**. Circuits for intelligent hardware control taking advantage of this knowledge, aiming to increase the information content. The infrastructure built to run several applications of intelligent microscopy simultaneously was called LifeXplorer, hinting at the final aim of automating science and the vision of the computer becoming a scientist.

Just as ARCO is supposed to be transferred to other science fields outside of life sciences, the experiment set-up based on the LifeXplorer platform is not limited to use in microscopy environments only. In contrast, the hardware platform for the experiment set-up plays a minor role. To efficiently and effectively run the developed ARCO applications however the LifeXplorer was enhanced with its own LifeXplorer MMS (Microscopy Managing System).

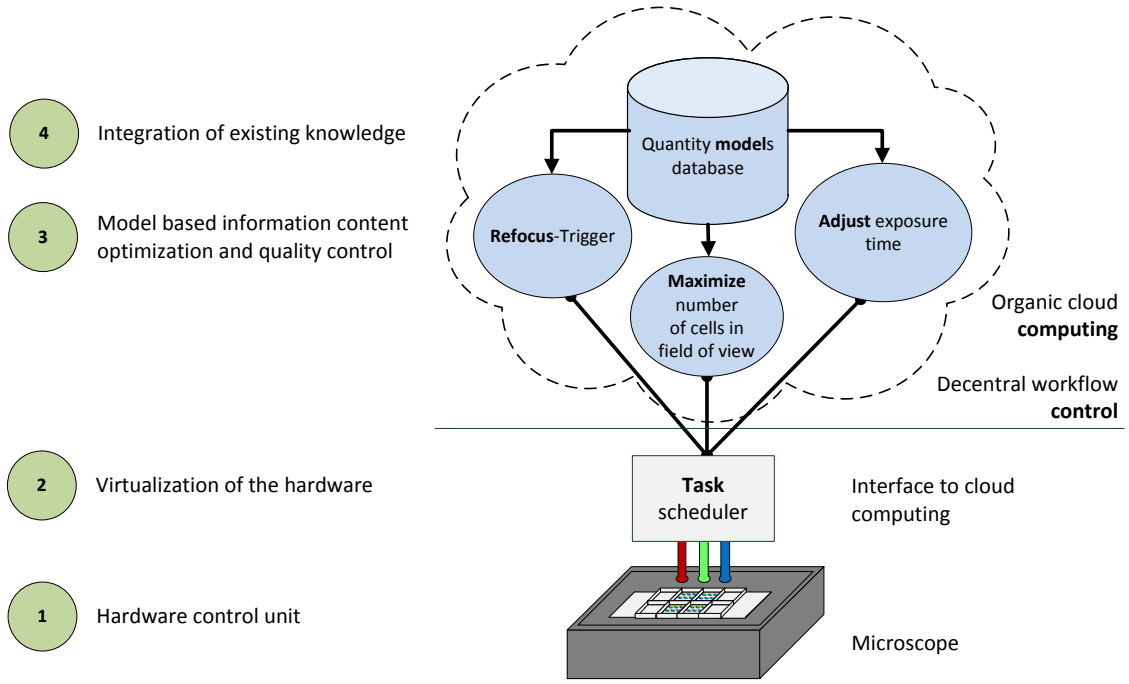


Figure 2 LifeXplorer architecture overview. The microscope hardware control unit (1) is virtualized by a scheduler (2) unit which organizes both hardware control commands to run imaging workflows, and data processing tasks for feedback controlled microscopy. The scheduling is based on organic computing approaches to be able to dynamically adapt to the time variant demands of the imaging workflow, as well as the data processing itself. The hardware and processing tasks are executed in a computing cloud on which different applications are run to optimize the information content and control the quality of imaging workflows (3). The integrated decision support systems take advantage of previous knowledge (4) that is formulated in mathematical models about the biological system and the scientific question. The data analysis can be adapted over time through events detected on run time. Circuits for intelligent hardware control can be run redundantly to ensure highly dependable workflows. Single circuits run optimizations and control the quality of all basic parameters of imaging workflows such as the light exposure, the three dimensional sampling point positions XYZ and the sampling rate.

1.3 Contributions

This dissertation investigates intelligent high-throughput microscopy applications for automated information densification and quality control. The environment in which this research took place was the upcoming field of quantitative biology, also called systems biology. The main contributions of the dissertation research are described below in the order they are presented in this thesis:

First, the main scientific contribution was to develop and evaluate an algorithm that measures the functional relation between a technical parameter and the information content. The algorithm gains a mathematical basis upon which a parameter of a system can be configured such that a scientific question can be answered, with optimized efforts and costs and an increased information content of an experiment. It helps biologists to observe biological systems more gently and specifically. This method called ARCO (Automated Result / Cost Optimization) was shown in the context of phototoxicity minimization and several other applications as follows.

Second, this thesis introduces the LifeXplorer platform for intelligent microscopy applications. Using a top-down approach, it is shown how complex data analysis and hardware control commands can be combined in one framework in order to be able to have a feedback controlled, dynamic, and adaptive imaging workflow.

Third, the ARCO algorithm was evaluated using the exposure time as the technical parameter to be minimized based on a scientific question. A method called ASEC (Application Specific Exposure Control) was derived using ARCO and led to a three- to sixfold minimization of photobleaching and phototoxicity effects in two different biological experiments.

Fourth, several applications to control the image quality and to enhance the information were evaluated.

1.4 Thesis Outline

The thesis is divided into five chapters, described as follows:

Chapter 1 gives an introduction to the topic of automating sciences. It seeks to automate the generation of insights based on ethical and technical aspects. The main research approach and question are described. All contributions of this work are listed.

Chapter 2 introduces the Automated Result / Cost Optimization (ARCO) algorithm and briefly shows practical applications and results. *Chapter 4* is directly linked to it, but goes into more detail. In addition to *Chapter 4*, a practical application of ARCO demonstrates in *Chapter 2* how to dynamically optimize data processing on run time.

Chapter 3 presents the motivation to develop a platform for ARCO applications called LifeXplorer. It is intended to give background information on the experiment set-up, such as the architecture of LifeXplorer, consisting mainly of an organic computing manager, so called workers, and a visual programming tool called LifeXplorer Intelligence Manager.

Chapter 4 presents practical applications derived from the ARCO algorithm in more detail. These applications aim to reduce costs and to increase the information content in two basic dimensions of microscopy: light exposure and position.

Chapter 5 concludes the present work. It reflects on the questions asked, and discusses possible future work, which would exploit the results of this dissertation and the potential of ARCO and LifeXplorer.

Chapter 2
Method: Automated Result / Cost Optimization

2.1 Abstract

Improvements in the capacity to perform and analyze time-resolved experiments are a fundamental challenge in quantitative biology, with automation providing the eventual solution. To that end the author developed an algorithm, termed Automated Result / Cost Optimization (ARCO), as a universally-applicable approach based on an operator-defined experiment result. ARCO increased the information density of high-resolution microscopy experiments while minimizing the associated costs, including the loss of biosensor sensitivity and phototoxicity. Several derivations of the ARCO algorithm provided automatic optimization of microscopy experiments via dynamic re-adjustments of parameters, including XY-positions and light exposure during run time. Furthermore, to integrate high-throughput microscopy with single-cell, high-sampling online data analysis, the author developed LifeXplorer, an adaptable processing and hardware control platform, here applied to the Olympus IX81 ScanR and Nikon Ti-E NIS Elements imaging systems. All applications of the ARCO algorithm were benchmarked against two intracellular events which are well known to be influenced by phototoxic stress; mitochondrial energetics and the reassembly of the Golgi. ARCO optimization significantly increased the information content and accuracy of experimental results, while reducing phototoxicity by several fold. The LifeXplorer platform integrates commonly-used tools (ImageJ, CellProfiler, ilastik, Matlab, etc.), allowing for adoption and development of new ARCO applications. The application of ARCO to both image acquisition and analysis is an important step towards automation and integration of microscopy and data analysis for the emergence of quantitative biology via computer vision.

2.2 Introduction

High-resolution imaging of subcellular, dynamic cell signaling events [16-19] coupled with data extraction and analysis allows for quantification, and improved hypothesis testing in biological research [20]. Increasing sampling throughput allows for the capture of heterogeneity within cell populations [21-23], and statistically describable phenotypes [21, 24]. Recent improvements in image analysis software allow for virtually unlimited flexibility in feature extraction, analysis and scalability [25]. However, implementation of computer vision, the integration of image acquisition, handling and analysis strongly depend on the concrete usage of existing tools and their parameterization. Individual expert knowledge is required therefore, and collaboration between specialized research groups determines the overall success of scientific experiments. Moreover, the biological value of the results is negatively influenced during acquisition and analysis, which significantly reduces the biological value of results. Automation of acquisition and analysis offers an attractive solution for reducing complexity and the requirement for specialization [26]. Integrative platforms, including Micropilot [4, 27], have been developed for offline and online data analysis. These approaches employ machine learning to extract and describe complex phenotypes. Machine learning approaches can utilize generated data to analyze factors underlying data generation, and are powerful tools emerging in the field of quantitative biology [12, 28-32]. Importantly, phototoxicity [5, 6] and photobleaching [7-9] are fundamental problems intrinsic to live-cell imaging, which reduce image quality and it is widely recognized that light exposure results in mitochondrial dysfunction [33, 34], and influences the pathway being explored. Reduced illumination has been achieved via optimized pixel-dwell time with specialized laser scanning microscopy [10, 11, 35-37], and through development of application-specific re-focusing algorithms [1-3]. However, to perform online data analysis for high-throughput microscopy data, a processing framework is necessary that is able to schedule computing tasks and generate feedback control signals for each measuring point individually, which at present does not exist. To

utilize ARCO, the LifeXplorer framework was therefore developed to integrate imaging, data handling, and machine-learning analysis. This chapter summarizes all major results of this thesis, as all applications are motivate by ARCO. Details and further applications can be found in chapter 4 Applications: of the ARCO algorithm.

2.3 Results

The conventional approach to imaging experiments relies on subjective parameter estimation (i.e. light exposure time often estimated by the human eye). Only after the data acquisition and analysis is performed, imaging optimization requirements are identified. To change this paradigm, the author included an additional degree of freedom with the ARCO algorithm. Describing the phenotype of interest prior to data acquisition, it is possible to integrate and optimize both parameter domains of data acquisition and analysis on runtime in three steps applying the ARCO algorithm (**Figure 1**). This method reduces experimental complexity for the operator.

Originally the ARCO algorithm was designed to only measure the non-trivial relations between light exposure and information content, and on this basis to minimize the phototoxicity associated with the imaging process. The author found that ARCO was furthermore able to automatically optimize other data acquisition and analysis parameters for quantitative live cell imaging, including XYZ-position, the magnification, the resolution, the sampling time, and the selection for optimal focusing and segmentation algorithms and parameters.

This chapter comprises an in-depth analysis of the ARCO algorithm to optimize the XY-position (**Supp. Figure 2a**), the light exposure (**Supp. Figure 2b**), the refocusing behavior (**Supp. Figure 6**) and to automatically select data analysis algorithms and parameters (**Supp. Figure 3**). In two biological systems, (1) the dynamic behavior of mitochondrial membrane potential and (2) the reassembly of Golgi apparatus, the author could measure significant

phototoxicity reduction and information densification using ARCO (**Figure 2**, **Suppl. Figure 4**). Time-lapse experiments consisted of four positions in four independent experiments for the mitochondrial membrane potential analysis, and 32 positions in four independent experiments for Golgi analysis. The ARCO implementation workflow is illustrated in **Suppl. Figure 1 and 4**.

Classically**,** the operator employs subjective light exposure configurations. In contrast, ARCO selects the optimal information content for a given data analysis and acquisition parameter, here nuclei and Golgi segmentation (**Suppl. Figure 2b**). As the information content is related to the light exposure (**Suppl. Figure 2b**), ARCO analyses different light exposures in order to select the lowest exposure time that is still suitable to detect 95% of the object area, which can be detected using the highest light exposure (**Suppl. Figure 2b+3**). Furthermore the best available data analysis method and configuration is selected to segment nuclei and Golgi (**Figure 1, Suppl. Figure 3+ 4**).

To run ARCO the author first defined the result R for both optical channels after the AutoExposure function was used to identify exposure time with the highest possible image dynamic (**Figure 1, Suppl. Figure 1**). The costs C were set equal to the exposure time. To define valid nuclei objects in the image, the pixel size of the segmented objects was defined to be in a range of 1000-6000 pixels and the Golgi apparatus in a range of 20-2000 pixels. For the Golgi complexes in addition the min. relative StdDev was set to 20% of the mean value. In the second step, the same image with 11 different exposure times (1 s down to 100ms in a 100ms step) was analyzed with multiple segmentation methods (global thresholding, CellProfiler [38] + ilastik [39], point source detector [40]) and configurations (threshold and percentile). Finally in the third step, surprisingly global thresholding was selected to maximize the result (object count) and the lowest possible exposure time (100ms) was identified to still segment 95% of the area correctly (**Suppl. Figure 3**). In contrast to the Golgi segmentation, the global thresholding segmentation pipeline was selected by ARCO to maximize the nuclei count (**Supp. Figure 5a-b**).

To further reduce phototoxicity, during runtime ARCO selects the action that minimizes the costs to get in-focus images. Below a quality threshold of 95%, in the case of the mitochondrial membrane potential measurements, a refocusing step is triggered and added to the scheduler by LifeXplorer (**Supp. Figure 6**). Above the quality threshold, the Z-position is readjusted by selecting the Z-plane of the imaged three planes stack with highest contrast value**.** Over eight hours of screening, only eight triggers on average every 76 min (**Supp. Figure 7)** were fired and caused a refocusing of an ARCO controlled spot. Using ARCO, the phototoxicity of the refocusing could be neglected.

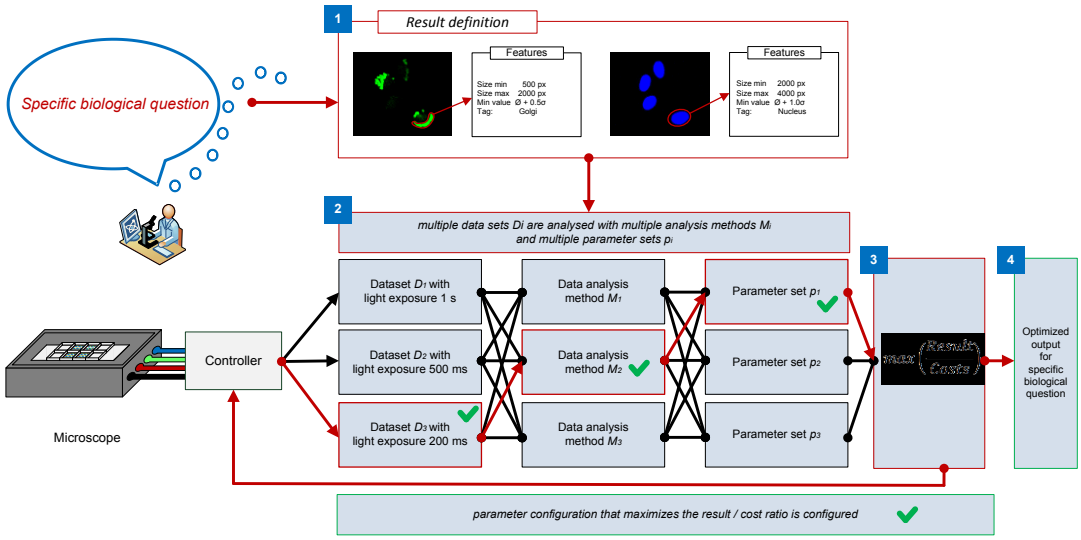


Figure 3 Schematic for the general three-step workflow of the Automated Result / Costs Optimization (ARCO) algorithm. It maximizes a specified result and minimizes the costs by evaluating the outcome with different data acquisition and analysis parameter settings (here illustrated with different light exposures). First, the result therefore has to be defined, using relevant existing knowledge about the measured quantities (such as geometry, brightness, etc.). Second, multiple analysis methods are run with multiple parameter sets for multiple acquisition parameters. Third, the parameter configuration for acquisition and analysis is configured, which maximizes the previously defined result (i.e. cell count) and minimizes the costs (i.e. light exposure). Finally, ARCO provides an optimized output for a specific biological question.

For the XY-position the ARCO algorithm was run with the same configuration as above, except for the cost function. The result R was defined as

the nuclei count and the costs C were set equal to one, ignoring the microscope's overhead to approach a non-equidistant position. Utilizing ARCO to maximize the information content by optimizing the XY-position (**Supp. Figure 5c-d**), the nuclei count was increased up to 220%.

To generalize ARCO for all experimental parameters, the author developed and formulated it according to equation 1, the maximization of the result / cost relation (RCR). Equation 1 formulates the general case, which considers all parameters of an experiment. Acquisition parameters used here include the XY-position, light exposure time, data analysis and focusing method. In Equation 2, the maximization of the overall RCR is formulated on the basis of the aggregated maximization of single parameters; assuming that parameters are independent and thus separate maximization is possible. Equation 3 represents the actual basis for the ARCO approach shown in this chapter. For different parameter configurations the result $\mathrm{R(p_i)}$ is computed. A priori a cost function $\mathrm{C(p_i)}$ for a parameter is formulated. Finally the parameter set with the highest RCR can be set (**Figure 1, Suppl. Figure 4**).

$$(\mathrm{p_0}, \mathrm{p_1}, .., \mathrm{p_i}) \rightarrow \max\left(\frac{\mathrm{R(p_0}, \mathrm{p_1}, .., \mathrm{p_i})}{\mathrm{C(p_0}, \mathrm{p_1}, \ldots, \mathrm{p_i})}\right) = \max(\mathrm{RCR}) \qquad \text{(Eq-1)}$$

$\mathrm{p_i}$ parameter;
RCR result $-$ *cost* ratio;

Parameter reduction assuming the parameters can be separated:

$$(p_o, p_1, \ldots, p_i) \rightarrow \sum_i \max\left(\frac{R(p_i)}{C(p_i)}\right) = \max(RCR) \qquad \text{(Eq-2)}$$

$$p_i \rightarrow \max\left(\frac{R(p_i)}{C(p_i)}\right) = \max(RCR_i(p_i)) \qquad \text{(Eq-3)}$$

Evaluation of ARCO with the LifeXplorer framework

Implementing a testable ARCO algorithm required an integrative data processing and hardware control framework, scalable for multipoint experiments and high-throughput feedback-controlled microscopy. LifeXplorer was therefore developed. It allows for desktop PC usage, supports redundancy and thus high availability for less reliable processing nodes that may crash or undergo delays.

LifeXplorer is designed as an open hardware interface to any microscopy system and allows ARCO implementation with standard analysis tools such as ImageJ [41] , CellProfiler [38], ilastik [39], Matlab and Mathematica. LifeXplorer integrates new data analysis methods to the ARCO workflows during runtime. It is able to run computing tasks for each measuring point individually on different computing nodes. Computing resources are virtualized and the processing controller automatically distributes tasks by the means of self-x and organic computing approaches [42]. In all, the LifeXplorer framework incorporates application specific exposure control (ASEC, **Suppl. Figure 4**), the automated algorithm and parameter selection (AAPS, **Figure 1** and **Suppl. Figure 5**), optimal positioning of the observed region (OSAPI, **Suppl. Figure 2a**), and a stack-applicable refocusing trigger (RFT, for individual description see **Suppl. Methods**). To enable adoption, LifeXplorer was designed to be compatible with standard interfaces, and was tested here with the Nikon software Nis-E, which controls advanced Nikon microscopes, and via a custom solution controlling the Olympus ScanR microscopy system. LifeXplorer is open source based on .NET and can easily be customized. To interface with Nikon microscopy systems, a macro is included into the imaging workflow that communicates with LifeXplorer. For the Olympus ScanR system, custom software was written to utilize ScanR hardware drivers. LifeXplorer can easily be connected to microscope systems and run ARCO applications by simply exchanging basic imaging commands and macros.

Optimized time-lapse imaging of mitochondria bioenergetics

Phototoxicity [5, 6] and photobleaching [7-9] are fundamental problems intrinsic to live-cell imaging, which reduce image quality and it is widely recognized that light exposure results in mitochondrial dysfunction [33, 34]. To address this problem the author performed experiments in HeLa cells loaded with TMRM (50 nM), imaging with the Olympus IX81 at 20x magnification for a period of 8 hours, at 30 second increments. ARCO was implemented to both optimize exposure time (ASEC) and optimize sample size through optimized XY-positioning of the measuring points (OSAPI, for individual description see Supplementary Methods). In addition, the refocusing trigger (RFT, for individual description see Supplementary Methods) was applied to avoid unnecessary image acquisition and thereby increased phototoxicity. Phototoxicity from TMRM excitation induces intra-cellular oxidative stress, propagating mitochondrial depolarization and mitochondrial ROS generation [33, 34, 43, 44]. In the absence of ARCO implementation, rapid loss of mitochondrial potential within the cellular population was detected between 1 and 2 hours of imaging. With ARCO-optimized light exposure, no sudden loss of mitochondrial membrane potential occurred, and a linear decrease over 8 hours can be attributed (**Figure 2a**) to dye leakage and photobleaching. Thus at the time, ARCO achieved a 3 to 4-fold reduction in the impact of phototoxicity to 50% depolarization and total depolarization, respectively (**Figure 2b**). Furthermore, total intensity was significantly maintained using ARCO, demonstrating decreased photobleaching.

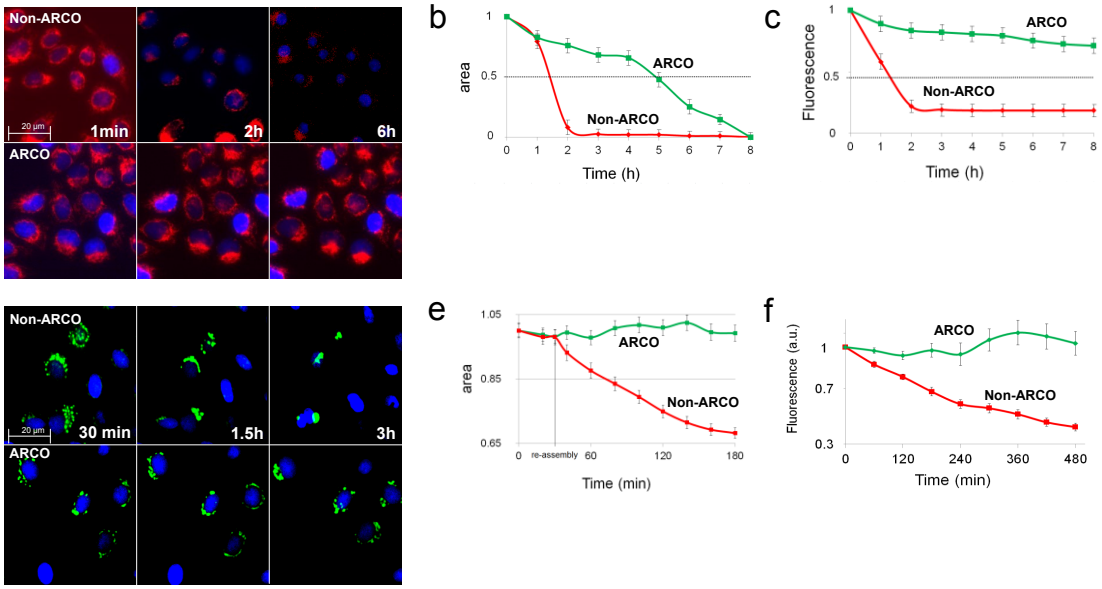


Figure 4 Optimized time-lapse imaging using ARCO by reducing phototoxicity during imaging of mitochondrial bioenergetics and Golgi reformation. (**a**) ARCO optimization for time-lapse imaging of mitochondrial membrane potential. The difference in phototoxicity between ARCO and non-ARCO can clearly be seen in projects of series of 3D stacks of HeLa cells with the electrophoretic mitochondrial membrane potential reporter TMRM (50 nM). (**b**) In response to continuous imaging, mitochondrial membrane potential is lost, resulting in rapid decrease in mitochondrial fluorescence (red line). The phototoxicity curve demonstrates that the phototoxicity rate is reduced three- to fourfold by ARCO (green line). (**c**) Photobleaching effects of time-lapse imaging can be detected and minimized three to fourfold. (**d**) Optimization for time-lapse imaging of Golgi reformation. The difference in phototoxicity between ASEC and non-ASEC can be seen in time-lapse imaging (every 2 minutes) of NRK cells expressing Golgi-associated GalT-CFP. (**e**) Under normal conditions, the Golgi appears as perinuclear stacks. Treatment with BFA (5 mL) results in fragmented, dispersed Golgi. During the reassembly, phototoxicity was detected after 1 hour by cell rounding for user-defined exposure time but not with the application of ASEC. Cell rounding was measured by the decrease in relative area (%) occupied by the nucleus. (**f**) After 1 hour of imaging, the GalT-CFP intensity is already bleached 25% in the user-defined exposure while no bleaching was observed with ASEC during 6 hours. Fluorescence was normalized to the intensity at the beginning of the experiment.

Optimized time-lapse imaging of Golgi reassembly

Golgi reassembly following treatment with the protein transport inhibitor Brefeldin A (BFA) can be used to determine regulatory roles of specific proteins [43]. This approach requires the use of a fusion of β-1,4-galactosyltransferase to cyan fluorescent protein (GalT-CFP) to detect Golgi reassembly over time based by quantifying juxtanuclear GalT–CFP localization following washout of BFA. Accurate quantification of Golgi reassembly requires that the process is not influenced by phototoxicity and that bleaching does not result in bias of measurement (e.g. underestimation of intensity). To that end, the author applied ARCO in order to increase information content through shorter imaging time intervals during both disassembly in presence of BFA and reassembly after washout. Two illumination schemes were run in parallel. Exposure time was calculated by either (i) the standard "auto exposure" function present in all commercial imaging software packages or (ii) ASEC-based optimization of Golgi segmentation in an exposure series (**Figure 2d**). The auto-exposure function increased the exposure time from the initial selected 100 ms to 1 second, improving the signal-to-noise and image quality (Figure 4D, upper panel), but more importantly induced rapid bleaching of GalT–CFP (**Figure 2f**) and phototoxicity, as measured by rounding of the nucleus within 60 minutes, 30 minutes following BFA washout (**Figure 2e**).

2.4 Discussion

Here the added value of the Automated Result / Cost Optimization (ARCO) algorithm is benchmarked to standard experimental approaches using existing microscopy platforms. Phototoxicity [5, 6] and photobleaching [7-9] are fundamental problems intrinsic to live-cell imaging, reducing image quality. It is widely recognized that light exposure results in mitochondrial dysfunction [33, 34]. Previously reduced illumination has been achieved via optimized pixel-dwell time with specialized laser scanning microscopy [10, 11, 35-37], and through development of application-specific refocusing algorithms [1-3]. Integrative

platforms, including Micropilot [4, 27], employ machine learning to extract and describe complex phenotypes. Machine learning approaches can utilize generated data to analyze factors underlying data generation, and are emerging tools in the field of quantitative biology [12, 28-32].

ARCO optimized the quantitative result value (object count and segmented area) of the imaging up to twofold (**Supplementary Figure 5**), while minimizing the costs of the acquisition process on the observation. This was demonstrated by a measurable decrease in phototoxic effect on mitochondrial bioenergetics and Golgi reformation. Phototoxicity was reduced 3-6 fold in both experiments. Thus artifacts stemming from the imaging process were almost eliminated (**Figure 2**).

Importantly, ARCO implemented in LifeXplorer is able to optimize parameters dynamically during run time and therefore makes costly trials of sequential data acquisition and evaluation obsolete. The decisions based on ARCO might be non-intuitive to the inexperienced user, as in the example of low signal to noise images resulting from illumination optimization (Figure 4). This is because the algorithm optimizes quantitatively for the scientific question rather than for bright images with a high dynamic range as in the case of the user.

Decisions to optimize the imaging workflow during runtime can be based on extended knowledge by using precise physical and biological models. Having now established a general method for how to optimize the efficiency and effectiveness of experiments, the ARCO algorithm can be applied to additional experimental parameters, forming a critical component of hypothesis testing and validation. The author started to apply and evaluate ARCO to select the best focusing algorithm for a specific experiment on run time, to refocus during time-lapse only if necessary, and to find a suitable sampling time for each channel individually. In a pre-scanning step the amount of relevant cells can be maximized by ARCO by modifying the measuring point based on a model of the phenotypes of interest. Further concrete imaging issues can be addressed as was shown in chapter 4 Future algorithms can be annotated on runtime for which experiment specifications they worked best with which parameter-sets. This finally can lead

to an evolutionary network, which supports imaging and data analysis optimization based on your scientific question and experiment specifications.

The automated result / cost optimization approach described here provides a powerful and adaptable approach to optimize data acquisition and analysis on runtime and can be applied to other experimental parameters such as focusing algorithm, magnification, resolution, sampling time and stacking.

2.5 Material and Methods

Hela cells were plated at 40000 cells per well in µ-Slide 8 well microscopy slide (ibidi, Martinsried, Germany) one day prior to the experiment. For analysis of mitochondrial depolarization, cells were stained with Tetramethyl Rhodamine Methyl Ester (TMRM) at a final concentration of 10n M for 30 minutes at 37°C, 5% CO_2. Cell nuclei were visualized through DNA staining with Hoechst 53342, at a final concentration of 1 µg/µl, incubation at 37°C, 5% CO_2 for 30 minutes. Live cell imaging was performed in full medium (DMEM, 10%FBS). NRK cells stably expressing GalT-CFP (NRK-GalT-CFP) were a gift from the Starkuviene lab and cultured as described [43]. Cells were plated in ibidi 8 well microscopy slide one day prior to imaging. The GalT-CFP redistribution assays was performed as described [4]. In short, cells were stained with Hoechst 53342 for 15 minutes prior to acquisition and after multiple positions per well were defined in the software, 5 µg/mL of BFA and 0.1 mg/mL of cycloheximide were added. Timelapse images were acquired every 2 minutes over 30 min. After washing cells 3 times with the preheated growth medium, time-lapse imaging of the same positions was resumed for 3 hours. Cycloheximide was kept in all solutions during Golgi reassembly to prevent de-novo synthesis of GalT-CFP.

The microscopy setups consisted of two different motorized inverted microscopes with wide field fluorescence. The first is an Olympus IX81, a frame grabber (Matrox Meteor-II) and a CCD camera (Hamamatsu C9100-02). Two objectives were used, a 10x objective (Olympus UPlanFL, NA 0.3) and a 40x

objective (Olympus LCPlanF1, NA 0.6). The second setup was a Nikon Ti-E with automated table and perfect focus system operated with Nis-Elements AR 4.1, using a 20x Plan Fluor objective (Nikon, NA 0.75) and an interline transfer CCD camera (Clara, Andor). The metalhalide illumination (Nikon Intensilight) was attenuated to 25%, rather than using low exposure times, which might then be shorter than the reaction time of the epifluorescence shutter (Sutter 10-2, Sutter Instruments), gating exposure. Initial exposure was estimated by using the auto exposure function based on saturation of pixels. This value was reduced by 40% to avoid potential saturation of pixels at other XY-positions. Microscopy computers were connected with the LifeXplorer software through internet access for online image analysis.

2.6 Supplementary Methods

2.6.1 Application specific exposure control (ASEC)

The first application of ARCO was to reduce phototoxicity in each channel. Based on Equation 1, the initial exposure time is chosen by the operator as the start value. The exposure time is automatically increased until saturation. The exposure time then is decreased incrementally and the segmentation efficiency is calculated each time, in this case to segment mitochondrial membrane potential or the Golgi complexes. Finally the minimal exposure time is taken where 95% segmentation efficiency is still possible.

2.6.2 Optimized sampling point identification (OSAPI)

Within the autofocusing step, images are taken with a low 10x magnification. The optimized sampling point for the higher magnification is set to the center of the region of interest with the maximal number of relevant cells. To prove the

method, nuclei were segmented using CellProfiler [45] and ilastik [39] that are plugged into the LifeXplorer control logic. The optimized sampling point identification led to a doubled information density in the case of 10,000 cells plated and four- to sixfold enhancement for 5,000 cells.

2.6.3 Refocusing Trigger (RFT)

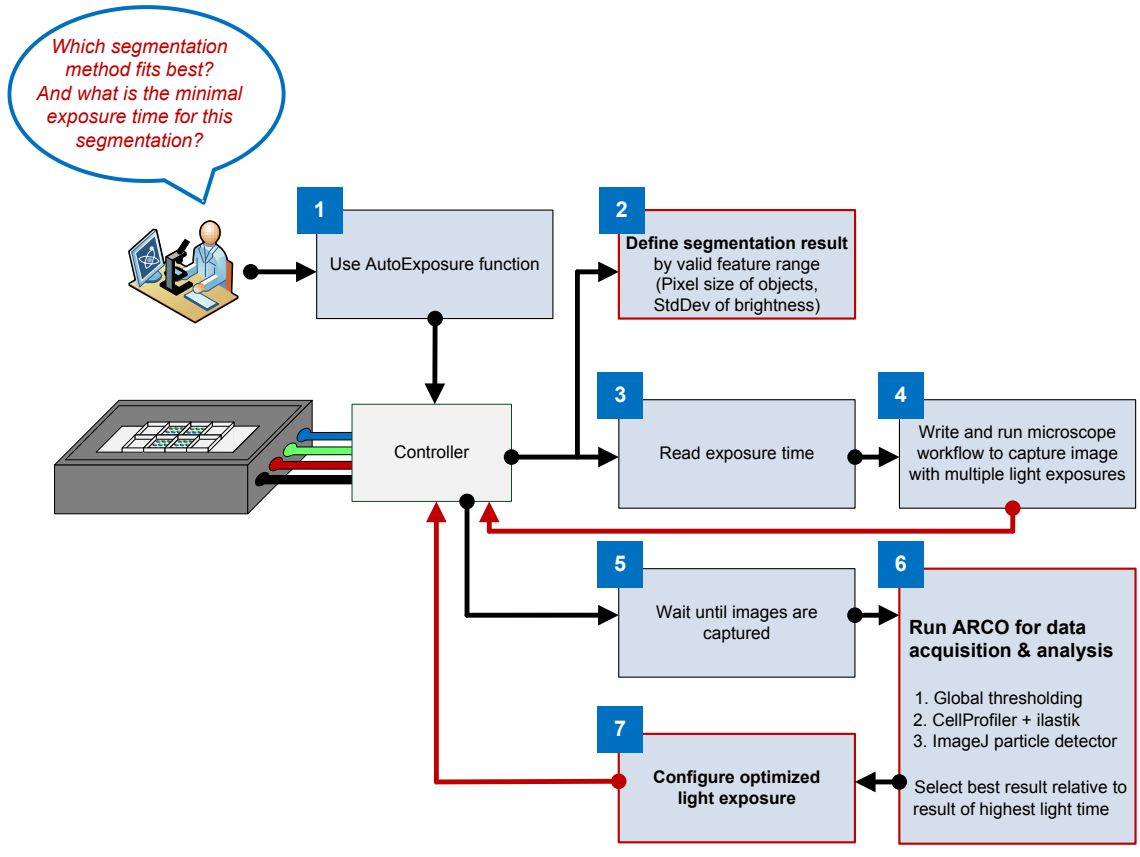
The refocusing trigger can be derived from ARCO as well. Initially after the focusing step, the contrast value of each sampling point is saved in the LifeXplorer computing cloud. The RFT logic then triggers the microscope to refocus a position if the quality of the image of a sampling point drops under a certain threshold. Including ARCO, this logic can be generalized by adding a cost function, which leads to a refocusing if it is more efficient to refocus a sampling point then losing information content, because the focus has drifted. Refocusing a position is time consuming and thus creates costs in terms of dimension time and phototoxicity. The information content in the z-dimension can be quantified i.e. by a contrast value since a scientific question can only be answered if the quality of the image is suitable to extract relevant information. If the contrast value of an image of a certain sampling point drops under a predefined minimum quality threshold, the costs for losing information content have to be taken into account, i.e. because the optical spacial resolution is decreased. In this case a cost function is dynamically created on run time: The RFT logic simulated both behaviors every time, refocusing a position or not, and calculated which behavior is more efficient based on then calculated ARCO efficiency. The phototoxicity could be reduced two to threefold by the refocusing trigger.

2.6.4 AAPS (Automated Algorithm and Parameter Selection)

ARCO is based on online data analysis. For data analysis, in this case segmentation, the author implemented an automated processing parameterization

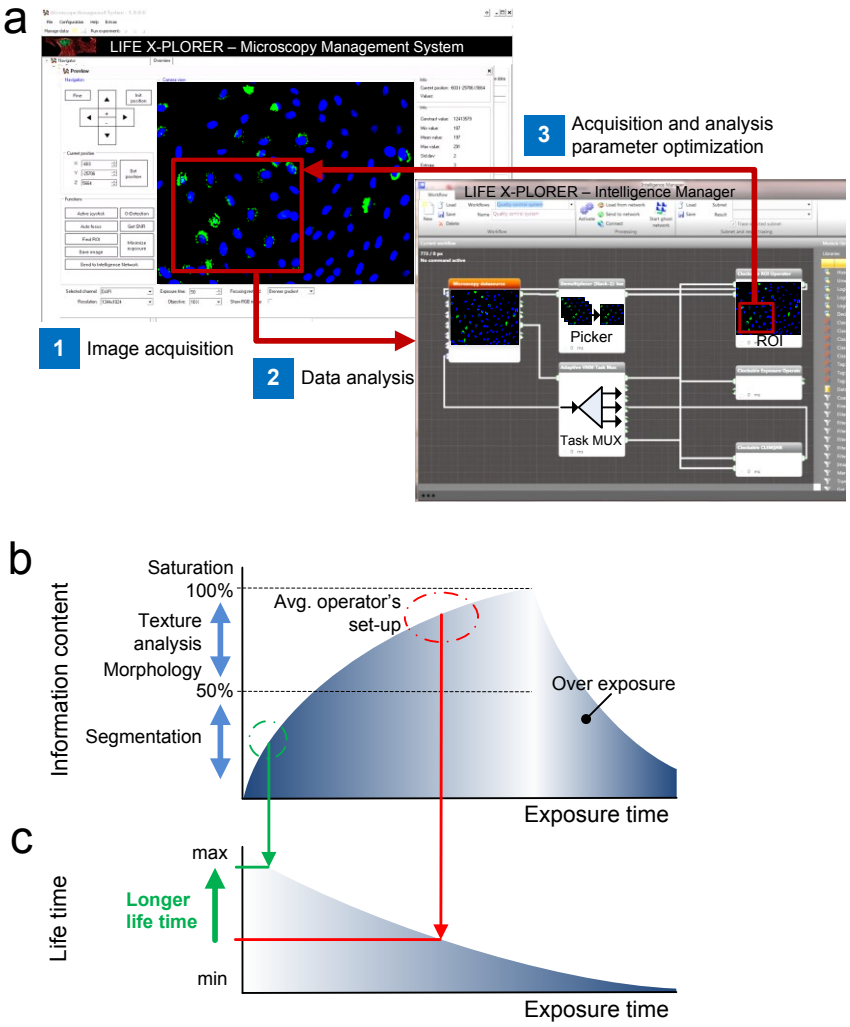
module for the LifeXplorer. It sweeps through a given parameter of the data analysis module and the parameter value with the highest amount of valid objects is then parameterized in the segmentation module. This approach ensures that one data analysis itself maximizes its outcome. Essential again is to define what this outcome quantitatively is before the experiment starts. Here the size of the cells has to be configured for an internal quality control after the segmentation.

2.7 Supplementary Figures



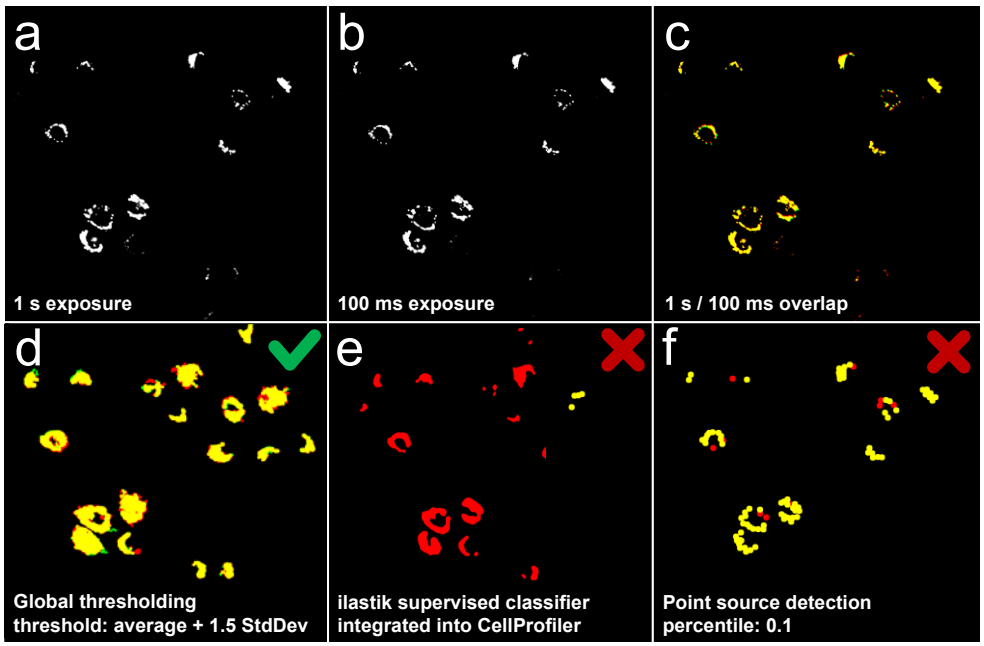
Suppl. Figure 1 ARCO experiment workflow. (**1**) Operator configures exposure time based on the existing AutoExposure function, maximizing the image dynamic. (**2**) Define the segmentation result by valid feature ranges for the pixel size of objects and the minimal StdDev of their brightness. (**3**) The exposure time is read out (i.e. 1000 ms) and (**4**) a workflow is created that captures the image with multiple exposure times within a range from 100 ms to the current exposure time in 100 ms steps. (**5**) LifeXplorer waits

until the microscope signals that all images are captured. (**6**) The images are read from the disk and three different segmentation pipelines are run against the images with multiple exposure times. First, a global thresholding within with 10 different thresholds within a range of average and max brightness are run. Second, a CellProfiler analysis pipeline is run using ilastik as a trainable classifier to detect Golgi complexes. Third, a particle tracker implementation for ImageJ is run with 11 different percentile configurations (between 0.1 and 2.0 with a step of 0.2). In every segmentation pipeline the previously defined result features are validated and the segmentation pipeline maximizing the result (object count) is selected. Based on this result the lowest exposure time is selected where 95% of the segmented area can still be detected. (**7**) The optimized exposure time is set.



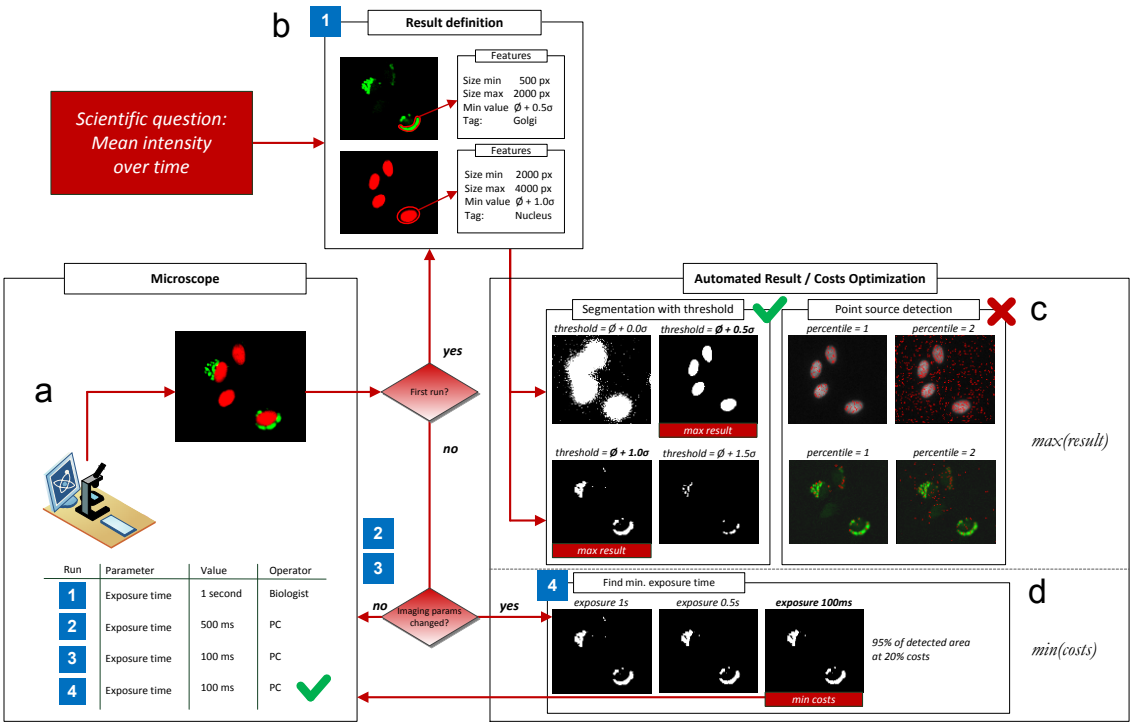
Suppl. Figure 2 ARCO application with LifeXplorer. (**a**) Intelligence Manager optimizing the XY-position to increase the cell count. After prefocusing with a lower

magnification, the region of interest is selected by maximizing the cell count. (**b**) Schematic illustrating the relation of light exposure to information content. As shown on the Y axis, different imaging functions need different levels of information content, i.e. segmentation works at lower information content whereas high-content feature extraction needs a high information content and thus higher light exposures. The operator, however, tends to choose the maximal dynamic range, which strongly decreases the survival time. (**c**) Schematic illustrating relation of information content and life time and phototoxicity [12]. Classically**,** intuitive operator light exposure configurations are used. ARCO in contrast selects the optimal information content for a given data analysis (**b**), in this case a segmentation analysis, which requires lower light exposure in the dynamic range, and thereby minimized phototoxicity.



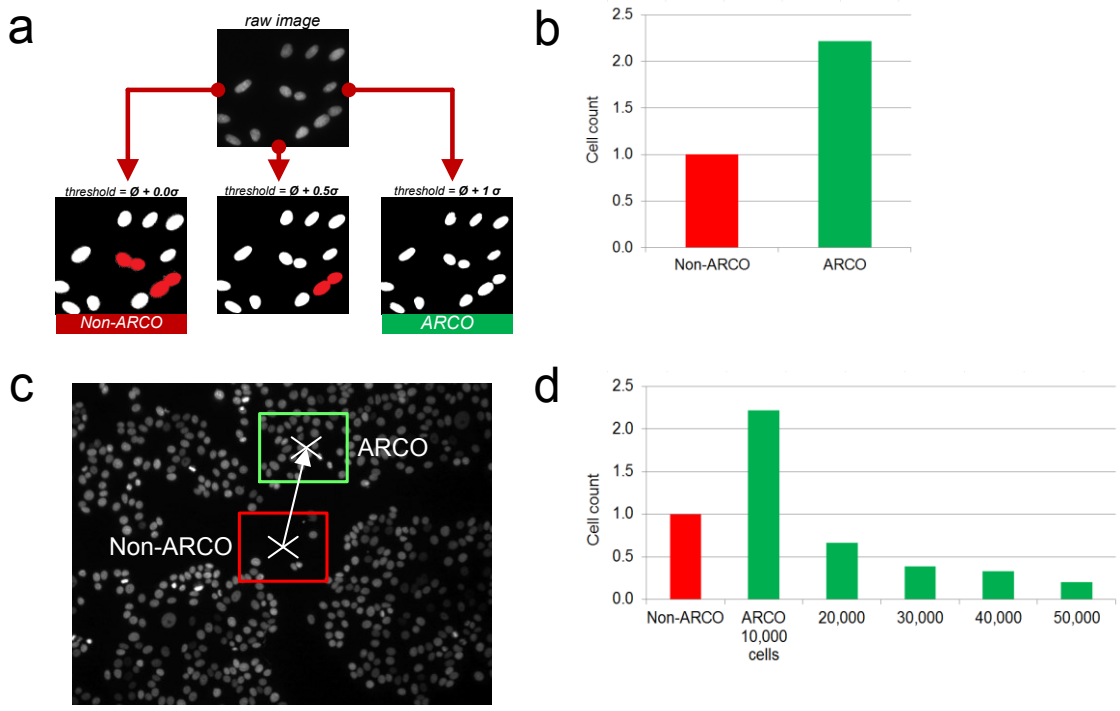
Suppl. Figure 3 Automated algorithm and parameter selection results for Golgi. (**a**) Binary image of Golgi complexes captured with 1 s light exposure. Dim Golgi complexes are not resolved. (**b**) Binary image of Golgi complexes imaged with 100 ms light exposure. (**c**) Overlap of (a) and (b). Yellow pixels are overlapping pixels, red and green pixels do not overlap. (**d**) Overlap map of Golgi segmentation using global thresholding, which was performed with 11 different threshold configurations: 0 - 2 StdDev + mean brightness, with a step of 0.2 StdDev. (**e**) Overlap map of Golgi segmentation using ilastik [39] integrated into CellProfiler [38] for Golgi classification. Only unique configuration was analyzed by ARCO. (**f**) Overlap map of Golgi segmentation using an

ImageJ [41] plugin for point source detection [40]. 10 configurations were analyzed by changing the percentile between 0.1 and 2.0 with a step of 0.2. ARCO analyses different light exposures to select the lowest exposure time that is still suitable to detect 95% of the area, which can be detected using the highest light exposure (Supplementary Figure 2b). Furthermore the best available data analysis is selected to segment nuclei and Golgi. ilastik integrated into the CellProfiler was selected by ARCO to maximize the number of valid Golgis (e). Being only interested in the average brightness over time, Golgi objects were defined with a pixel size range of 20-2000 pixels and a min. relative StdDev as 20% of the average value. Surprisingly, global thresholding with a threshold of mean + 1.5 StdDev performed best and was chosen by ARCO as it could classify the highest amount of Golgi complexes, being sensitive to a high dynamic brightness range.

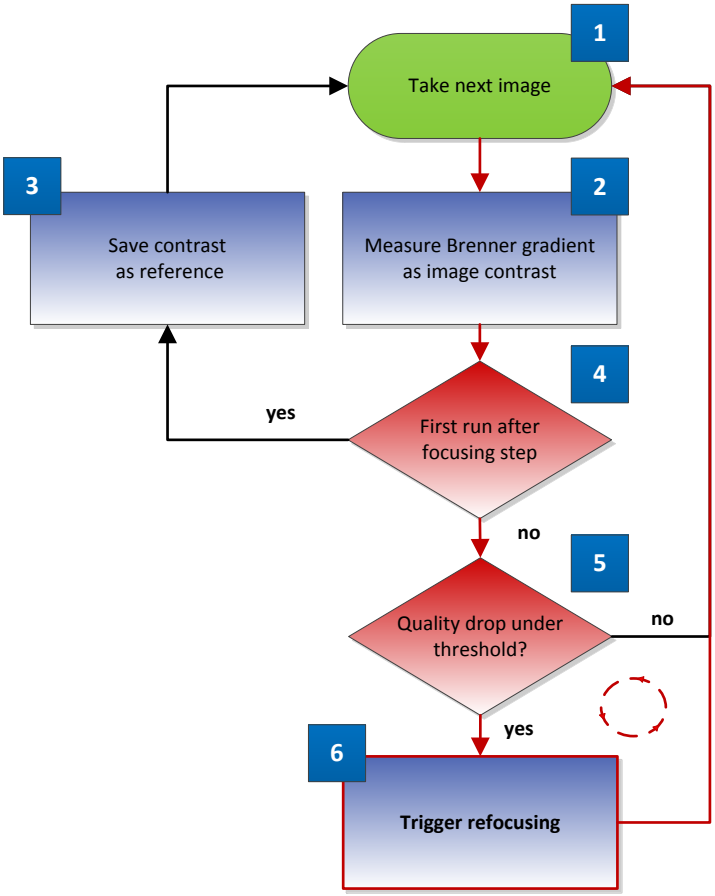


Suppl. Figure 4 Two separate application domains of ARCO, data analysis and data acquisition. (**a**) The microscope's parameters are changed automatically and the parameter configuration with the lowest costs and the highest result is set automatically. (**b**) The first, fundamental step for ARCO optimization requires result definition. Shown are example workflows for detection of the nuclei and Golgi complexes. Definitions for organelles are determined by size and average brightness features. (**c**) Different algorithms for segmentation are run with different parameter configurations. Based on the result definition, the segmentation results are filtered and only valid objects remain.

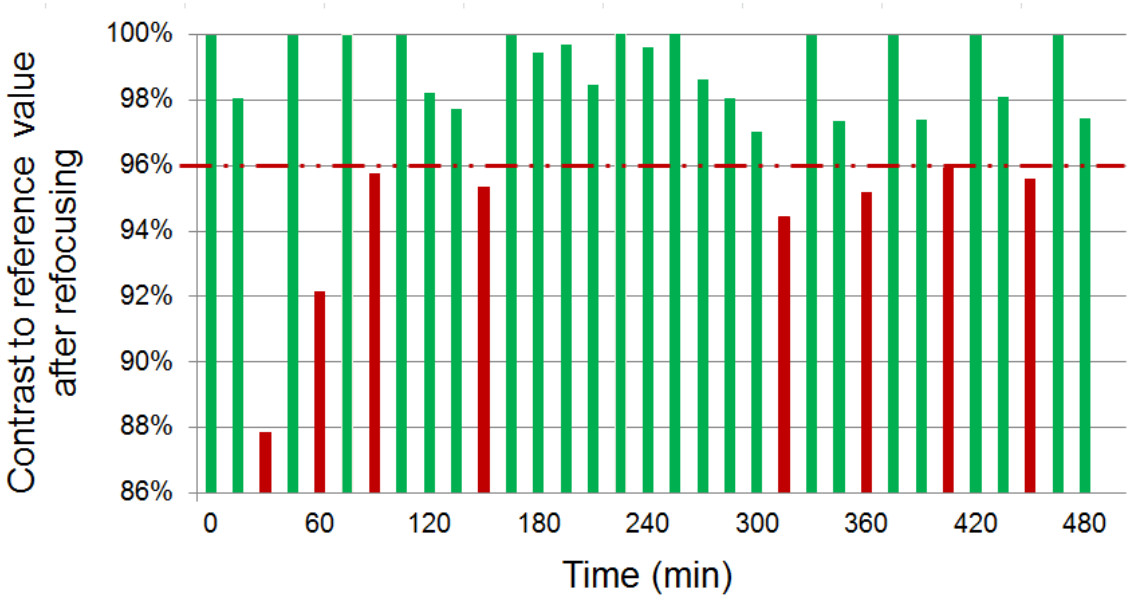
The algorithm and parameter configuration with the highest amount of valid objects is selected. (**d**) Based on a reference imaging position, the segmentation algorithm is applied to a range of exposure times, and the lowest exposure time with best result (i.e. 95% of the area is segmented) is used for data collection, for each time point.



Suppl. Figure 5 Experimental application of ARCO to optimize the number of cells detected. (**a**) Improvement of single nuclei detection. ARCO can be applied to maximize the nuclear detection, i.e. cell count, by selection from multiple segmentation algorithms, including global thresholding, supervised classification implemented in ilastik and CellProfiler [38], and a point source detector [40] and changing their parameters if possible. (**b**) ilastik integrated in CellProfiler performed best and could improve sensitivity by 40% compared to a standard segmentation pipeline with global thresholding using ImageJ. (**c**) Example image HeLa cells labeled with nuclear dye Hoechst (1 µl). After prefocusing with a 10x magnification, a 40x region of interest with the highest cell count is selected as the new image center. (**d**) For Hela cells plated at a density of 50 to 10,000 cells per well, the nuclei count was increased up to 220%.



Suppl. Figure 6 Refocusing trigger system. (**1**) An image is taken and (**2**) the Brenner gradient [46] is measured as the image contrast. If it is the first run after a focusing step, the contrast value is saved as a reference (**3**), else the reference value is compared with the current contrast value (**4**). If the contrast drops under a threshold of 95% (**5**), the system triggers a refocusing step (**6**). In the next run, the system is reinitialized with the new contrast value and it is saved as the reference value. Three modes are implemented: one trigger signal can trigger the refocusing of all spots, of a specific spot or all spots of one well are corrected by the Z-value different of one reference spot that needed to be refocused. ARCO costs are set to two for contrasts below 95% and to one above or equal to 95%.



Suppl. Figure 7 Contrast quality over time. The Y-axis is normalized to the contrast value measured after a refocusing step was performed. Where the relative contrast quality drops under a threshold of 96%, a refocusing trigger was necessary. After refocusing, the reference value is reinitialized and contrast measurements get close to 100% of the relative quality. Refocusing triggers are marked with red. Over eight hours of screening, the refocusing trigger was fired only eight times on an ARCO controlled spot. On average a trigger was fired only every 76 min. The ARCO decision support decided to perform a realignment of the Z-position based on the stack data in almost 99.4% of the cases, as the costs for the refocusing were set to two and the cost for stack based Z-position realignment to one. Sampling the mitochondrial membrane potential every 30 sec, the phototoxicity of the refocusing can be neglected when using ARCO.

Chapter 3
Implementation: the LifeXplorer platform

To show diverse ARCO applications, it was important to build an experimental setup with the ability to efficiently integrate them. Since this case study of ARCO is based on microscopy applications for life sciences, the main focus initially was to be able to build the LifeXplorer platform as a platform for intelligent microscopy applications. Intelligent microscopy is yet another word for feedback controlled microscopy. Having the possibility to process the imaged data online while the experiment is still running, and making decisions about the configuration of the workflow on run time, "intelligent" or rather feedback controlled microscopy applications are possible. In principle this option already exists by being able to plug-in external macros to an imaging workflow, as can be done in the latest versions of the microscopy managing software Nikon NIS-Elements. This standard interface to microscopy systems from Nikon was also used to provide ARCO applications for Nikon systems and to run biological experiments on Nikon systems feedback controlled by the LifeXplorer. Two different software components were built into the LifeXplorer platform: the microscopy control component called LifeXplorer Microscope Management System (MMS) and the LifeXplorer Cloud Computing Suite (CCS), consisting of an organic computing service and an intelligence manager. The first component has a very specific architecture which directly connects to a microscopy hardware platform, such as the ScanR from Olympus, based on an Olympus IX81 inverted motorized microscope. It is a simple imaging workflow management system, which can be connected to the LifeXplorer Cloud Services via SOA (Service Oriented Architecture) components. The microscope in this case becomes a service in the LifeXplorer cloud, which offers several basic functions for imaging. The LifeXplorer MMS has several advantages in comparison with standard interfaces, such as macros for Nikon systems. The main advantage is the full control over all functions. The microscope therefore could be integrated into the LifeXplorer platform using SOA components, and the overall architecture could

be built in exactly the way experimental set-up was needed for evaluating the potential of ARCO. It was found that no existing solution provided the necessary degrees of freedom to specifically implement what was necessary for our investigations. In the following paragraph, the state of the art is briefly described to give an overview of distributed systems, as the LifeXplorer system is a distributed system designed to be a scalable computing platform for high-throughput experiments such as high-throughput microscopy imaging applications.

After a short introduction of the motivation to create LifeXplorer as a platform for automating science, the state of the art is put into comparison to LifeXplorer. In chapter 3.2 Basic Technologies for Imaging and Data Processing relevant basics of the application context and the implementation of LifeXplorer is introduced briefly. The general architecture of the LifeXplorer is then described in 3.3 Software Architecture, along with all its relevant components. 3.3.1 Modules explains the main concept of abstracting hardware control and computing tasks and providing a unique interface for developers to add new functionality to LifeXplorer. In 3.3.2 Microscope Management System, the hardware control framework is explained briefly. The chapter 3.3.4 Intelligence Manager mainly shows examples of the graphical users interface where data analysis workflows and feedback control circuits can be setup. 3.3.5 Organic Computing explains the implementation of the organic computing functionality LifeXplorer is built on. For parallel and organic computing so called workers are necessary in LifeXplorer's architecture and are explained in 3.3.3 Workers. LifeXplorer was built to handle large amount of data and extract single object features over time on multiple measuring points. A database based the MS SQL Server therefore was created which helps to process statistics and is presented 3.3.6 Database. Finally LifeXplorer provides several plugins by default, as they were developed and used for the experiments and application of ARCO presented in this thesis. 3.3.7 Plugins for Image Processing and Hardware Control therefore gives a brief overview on the existing plugins for intelligent microscopy.

3.1 Introduction

Let us introduce this chapter by calling back into memory why it is relevant to automate science. And starting from this motivation to derive requirements for the concrete implementation for science automation presented in here. Good science is based on robust methods to get insights into life. Method optimization therefore is a key aim in science. Automation can help to optimize the method of insight generation. This is due to several reasons. The first reason especially in the field of life sciences is that, automation helps to manage complexity, which cannot easily be reduced only be optimizing scientific questions and experimental setup, since biology is always multidimensional and it is not yet clear how to reduce experimental complexity easily, without making use out of science automation. While manual observations and evaluations are very often necessary to have direct understanding of data and results, without automation errors sources and the variance of results can dramatically increase. Manual measurements and evaluations are always subjective. In addition as shown in science theory, i.e. by Karl Popper, the existing knowledge of us determines also the interpretation of results. As individuals have different experiences and knowledge, the results can heavily vary. Furthermore humans are limited in their senses, and make use out of technology became essential for modern science, focusing on small scales and high precision. Science however is aiming to create robust models about life and therefore needs reproducible results. Automating can essentially help to reach that aim, by the need to explicit and mathematical formulation of existing knowledge. This leads to unique basis for sharing and using knowledge. Finally automation can help to do model based experiments, meaning that the explicit knowledge formulation base is used to even drive experiments. Although we would like to avoid anything subjective and "human" in science and automation therefore is a suitable solution, human finally have to do the "brain work" and make use out of scientific results. Science however is complex and demanding. The more complexity is involved in an experiment, the harder it gets for humans to make it reproducible. Applying science automating therefore is a key approach to make scientific experiments and evaluation more

efficient and also effective. Both helps also to increase the motivation to perform experiments and explore life. The motivation enhancement finally also plays an essential role in science and tools for automation directly help to increase the human motivation to do science. Eventually automation has the potential to increase the speed of insight generation dramatically, while scientists are able to focus on their scientific question rather than spending time on the set-up and data analysis. The process of science automation is illustrated in **Figure 5**.

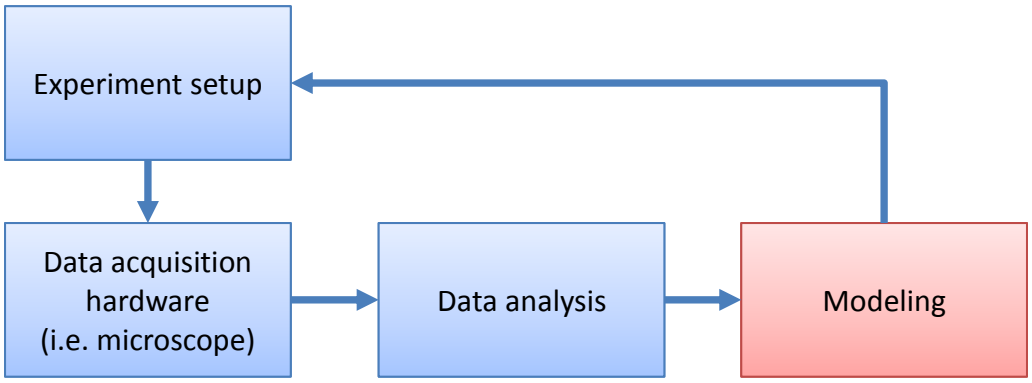


Figure 5 General process of insight generation in science

The usage of previous knowledge increases automation efficiency. The model cannot be considered as the end of the pipeline, but also as the start. Using formalized knowledge of this model for the process therefore is a key to science automation. Implementing this loop hardware needs to be coupled with an instrumentation that supports data analysis, the integration of a model and finally the power to control the hardware, based on gained knowledge.

Having gained this consciousness about what can science automation be about, we can derive what is necessary to efficiently integrate knowledge in order to integrate the insight generation loop above. First of all, a feedback control loop needs to be implemented. This implies that during runtime, data analysis and the usage of a model need to be executed (unsupervised in the case of full automation) to optimize the experimental setup and acquire optimized data, which again is analyzed. Optimizing parameters during runtime, however, needs a system that is able to handle the data acquired by the microscope or any other measurement system. The system to process the date therefore needs to scale for

different data rates. Data rates may even change over time, due to changes of the experimental parameters, like the sampling rate or the amount of list of sampling point. Reacting on biological events is time critical and therefore requires at least a soft real-time data analysis, which reliably provides a result in a given time frame, independently from variability of the processing environment. Creating feedback control logic for event based parameter change triggers or other parameter optimizations during runtime, due to efficiency and better motivation reasons, requires fast and easy integration of existing standards. Integration of existing standards such as ImageJ, CellProfiler, Matlab, etc. therefore was an important step towards fast and robust feedback circuit creation. Feedback circuits and data analysis workflows however can be complex and LifeXplorer should be filling a gap between specialized knowledge from image processing and statistics experts to biologist. Visual programming of the automation workflow therefore was in addition important to bring the added value of LifeXplorer and ARCO directly to the operator.

At the start of the project no tool was published by the scientific community which was able to meet the requirements mentioned above. At the end of the project however, another group developed a tool called Micropilot [47], which also serves as a platform for intelligent microscopy. Micropilot is based on LabView [48], however, which i.e. is used for the Olympus ScanR microscopy platform as well. LabView in contrast to LifeXplorer is able to handle multipoint experiments with redundancy and special binding behaviors. LifeXplorer explicitly was designed for dynamic parameters changes during time, such as changing sampling times or a dynamic list of sampling positions, which both directly influences the data rate and need dynamic resource allocation during run time. Micropilot in addition implemented a specific method for biological event triggering based on trained classification of phenotypes. This trigger activates an external hardware added to the microscope which can add a liquid to the sample, i.e. for event based drug treatment. In contrast to LifeXplorer it was not designed to actually change the acquisition workflow and all its parameters by providing sampling point specific processing nodes the ability to take decision on the workflow during runtime. LifeXplorer as well implements ARCO modules and is able to process

intermediate results of the ARCO algorithm in parallel. Depending on the configuration it can also bind modules to specific computing nodes, i.e. to nodes where GPUs (graphic processing units) are available and existing GPU code can be executed. Although Micropilot is a bundle for intelligent microscopy, which is based on LabView, it cannot be customized for a lot requirements mentioned above such as dynamic and high-throughput specific task scheduling. This is due to the fact that LabView is not open source and cannot freely be adapted. To meet the requirements above and evaluate novel methods of dynamic parameter optimization, LabView would have to be fully customizable, especially its parallel computing scheduler. Furthermore the aim of LifeXplorer is to also provide parallel computing based on a normal desktop PC network as it is available in every office environment. Facing this requirement, technical requirements such as redundancy in combination with sampling point specific parallelization and data binding are necessary, which was integrated into LifeXplorer as it will be explained in this chapter (s. the implementation for high availability through redundancy in **Figure 25**. Moreover LifeXplorer offers standard interfaces to visualize intermediate processing results, which is a relevant for proof of concept developments. LifeXplorer key requirement was to be able to have asynchronous workflows within the same environment that handles hardware control and processing tasks. While the microscope acquisition workflow is running, both synchronously (i.e. for focusing) and asynchronously (i.e. for a refocusing trigger) feedback are available. The processing and the acquisition workflow can be synchronized with built in methods for resynchronization of asynchronous signal. A scheduler was necessary that virtualizes the microscope hardware, so that multiple decision support systems running on different processing nodes, can add tasks to change the parameters of the acquisition workflow. LifeXplorer was designed to integrate other tools therefore, as single tools add value for the operator, but cannot meet the requirements for multipoint and dynamic parameter specific task scheduling. As Micropilot add value by its integrated image processing method to identify phenotypes, it can be integrated into LifeXplorer, yet not the other way around without losing the added values of scheduling and integrated ARCO modules which are built into LifeXplorer.

3.2 Basic Technologies for Imaging and Data Processing

The technologies described in the following pages are supposed to give a brief overview on relevant aspects of the actual architecture of the LifeXplorer, the microscopy platform it was connected to and the application context. Certainly this chapter is not intended to provide a complete overview on fundamentals and the state of all technologies involved. It is rather focused on a small introduction to digital imaging, microscopy, and distributed systems. LifeXplorer can run applications for feedback controlled microscopy in parallel, in order to control a hardware platform to optimize the outcome of experiments while reducing involved costs. General approaches are explained, as well as existing software technologies and frameworks which were used in order to build the LifeXplorer with the aim of being scalable in function, and to build a framework for automating science that can easily be used in the most relevant system environments.

3.2.1 Wide field microscopy

Focusing on the ARCO method and its practical application, an inverted fluorescence microscope was used as a hardware platform in order to demonstrate and evaluate the effects of ARCO within the environment of live cell experiments using different, automated wide field microscopes. Next to wide field microscopes, there are different microscopes available for different applications, such as laser scanning microscopes i.e. as confocal microscopes [49, 50], or more lately developments such as multi-photon microscopes [51]. Laser scanning microscopes create a spatial image of the object by sequentially sampling each pixel, which leads to reduced temporal resolutions. Applying multi-photon microscopy could combine both a reduced phototoxicity and a high depth penetration [52]. The advantage of wide field microscopy however is to create an image of one layer

with a single illumination. Excitation of many object pixels in parallel cannot be reached with laser scanning microscopy.

In **Figure 6** a compound microscope with two lenses (also see [53]) is illustrated. A whole area of an object is projected to the camera chip. Only the in-focus plane is captured with high contrast. As the point spread function of a wide field microscope has a low z-precision, other layers of the object are also projected to the camera chip and reduce the contrast of the in-focus signal. Because of a smeared point spread function, a whole volume of the object is finally aggregated on the camera chip, which reduces the spatial resolution.

The magnification M can be calculated with:

$$M = \frac{f_1}{f_2} \qquad \text{(Eq-4)}$$

f_1 is the focal distance of the objective and $f_2$the focal distance of the second lens. Point in the object plane with the distance d will have a distance of M*d in the image plane. Because the field of view of a microscope system is limited by the geometrical composition of the lenses, their focal distance and the refraction index of the material, the so called numerical aperture of a lens is relevant to identify the maximum field of view for a given magnification.

Numerical aperture (NA):

$$NA = n \sin \Theta \qquad \text{(Eq-5)}$$

whereas n is the refraction index of the medium between the lens and the object (i.e. air or oil) and Θ the angle of aperture which can still be captured in the light path. For quantification it is relevant to know the maximum revolution of a microscope system. The revolution is determined by the ability to distinguish between two separate object points in the image layer.

$$d' = d * M = 1.22 \frac{\lambda}{NA} \qquad \text{(Eq-6)}$$

λ is the wavelength of light, which has to be considered. d' is called the lateral resolution of a microscope. The pixel size of the camera chip determines the revolution, once it is greater than the lateral resolution. **Figure 7** illustrates the Epi-illumination based on the Koehler illumination. Excitation; and emission light can be separated with specialized filters, called beam splitters which are based on dichroic filters. This approach is commonly used in fluorescence microscopy, as fluorophores emit light after being excited in a different bandwidth as the wavelength of the excitation light. As illustrated in **Figure 8** and mentioned in [54] "August Koehler introduced a new method of illumination that greatly improved image quality and revolutionized light microscope design.[…]He introduced a collector lens for the lamp and used it to focus the image of the lamp on the front aperture of the condenser. A luminous field stop (the field diaphragm) was then focused on the specimen with the condenser focus control. The method provided bright, even illumination, and fixed the positions of the focal planes of the microscope optics. In later years […] fluorescence microscopy with epi-illumination would utilize and be critically dependent on the action of the collector lens, the field diaphragm, and the presence of fixed conjugate focal planes that are inherent to Koehler's method of illumination."

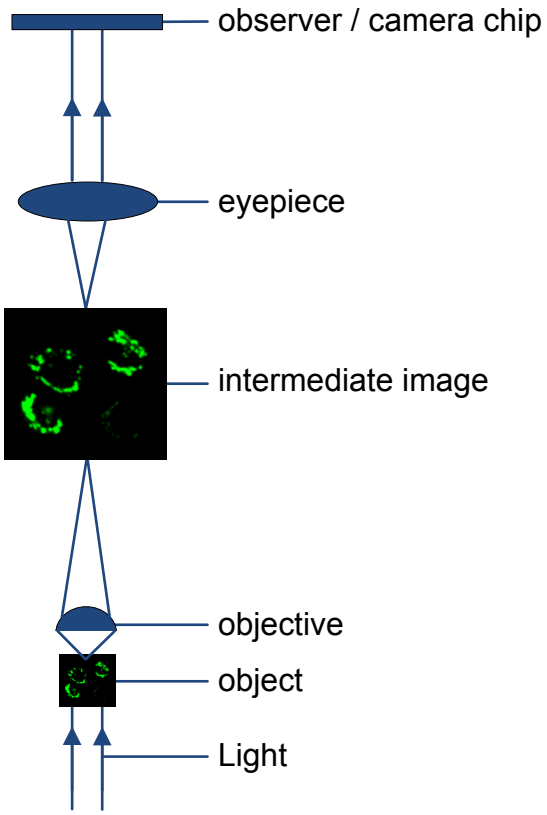


Figure 6 Compound microscope with two lenses.

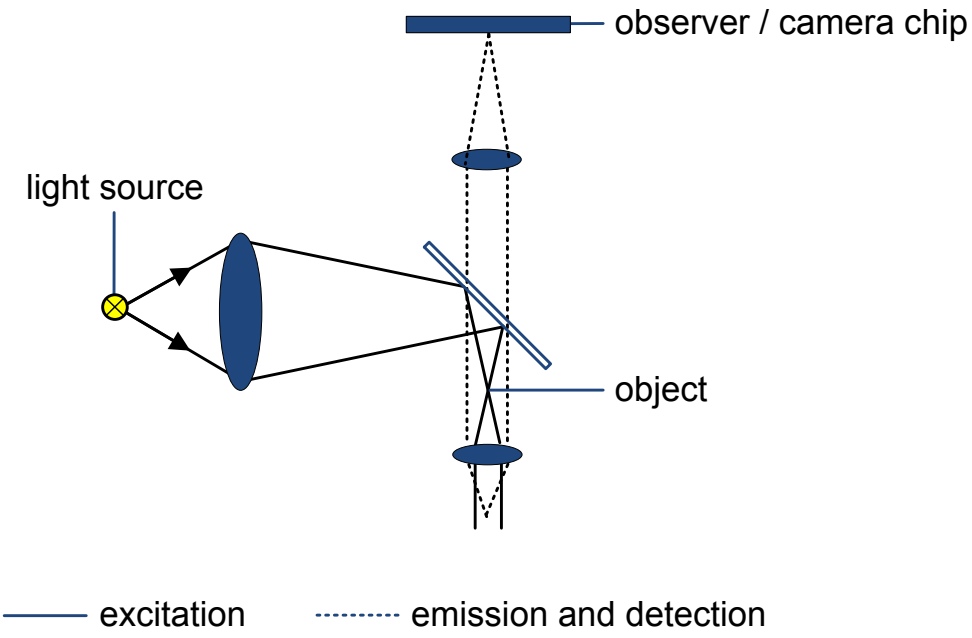


Figure 7 Epi-illumination with Koehler illumination. Image adapted from [55].

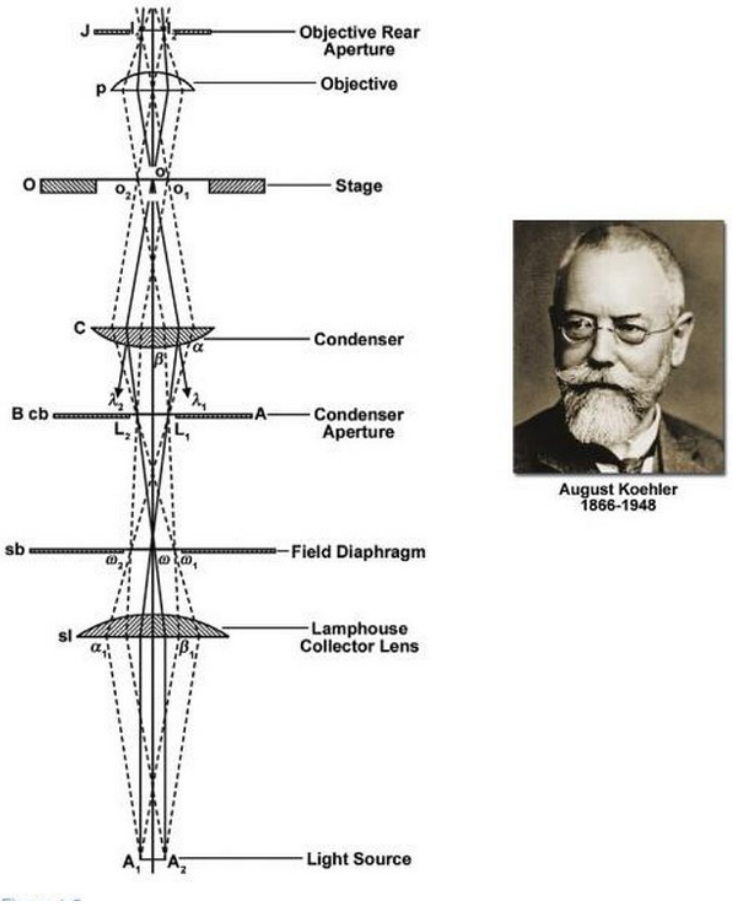


Figure 8 "August Koehler introduced a new method of illumination that greatly improved image quality and revolutionized light microscope design. Koehler introduced the system in 1893 while he was a university student and instructor at the Zoological Institute in Giessen, Germany, where he performed photomicrography for taxonomic studies on limpets. Using the traditional methods of critical illumination, the glowing mantle of a gas lamp was focused directly on the specimen with the condenser, but the images were unevenly illuminated and dim, making them unsuitable for photography using slow-speed emulsions. Koehler's solution was to reinvent the illumination scheme. He introduced a collector lens for the lamp and used it to focus the image of the lamp on the front aperture of the condenser. A luminous field stop (the field diaphragm) was then focused on the specimen with the condenser focus control. The method provided bright, even illumination, and fixed the positions of the focal planes of the microscope optics. In later years [...] fluorescence microscopy with epi-illumination would utilize and be critically dependent on the action of the collector lens, the field diaphragm, and the presence of fixed conjugate focal planes that are inherent to Koehler's method of illumination." [54] Images are adapted from [54].

3.2.2 Fluorescence microscopy

Fluorescence microscopy [56-59] became an essential tool for modern biology. Imaging of subcellular, dynamic cell signaling events [16-19] coupled with data extraction and analysis allows for quantification and improved hypothesis testing in biological research [20]. Quantitative imaging [16] today is a key technology to create models which are suitable for quantitative reproducibility. Finally, modern life sciences aim to receive insights into biological systems and their dynamics [60-62] over time, being able to take influence in a reproducible and predictable way. The physical effect of fluorescence offers a tool for microscopy where dyes [63] or the tissue itself are excited with light of a certain bandwidth, and nanoseconds later emit light in a different bandwidth. Excitation and emission light can be separated with specialized filters, called beam splitter based on dichroic filters as it is illustrated in **Figure 7** and **Figure 9**. So called fluorochromes can directly be employed for fluorescence microscopy, whereas other tissues need be labeled with dyes. In both cases intracellular events can be measured, and even single molecules can be detected applying ultra-high resolution imaging [64-71]. The resolution speckles can be detected using standard camera sensors. Fluorescence microscopes commonly offer a motorized XYZ stage, motorized filter cube revolver, and a motorized objective revolver. For dye specific light excitation, light sources offer filter wheels, which filter white light into single excitation bands. For laser microscopes, different laser light sources can be used and injected into the excitation light path. Being able to excite different dyes sequentially or even in parallel, multispectral biological readouts can be performed automatically for multiple positions. Automated microscopy [72-74] was a base technology for this thesis to implement and evaluate novel methods of automation and intelligent microscopy with the

LifeXplorer platform. In **Figure 9** a state of the art inverted fluorescence microscope is illustrated. Reflected light illumination is used together with beam splitters to separate fluorescence excitation and emission light.

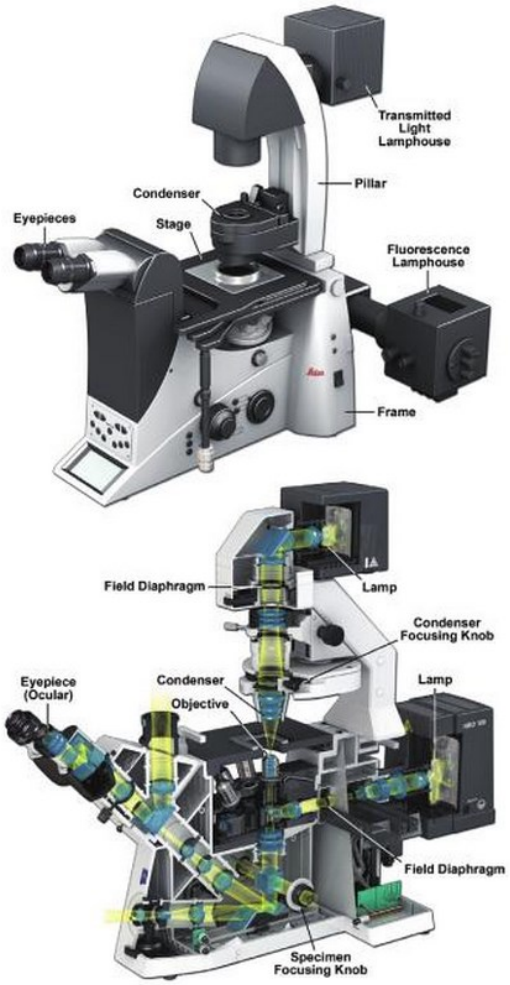


Figure 9 "The research light microscopy with inverted stand. As in upright designs, two lamps provide transmitted and reflected light illumination. Note the location of the knobs for the specimen and condenser lens focus adjustments, which are often in different locations on inverted microscopes. Also note the positions of two variable iris diaphragms: the field diaphragm near the illuminator, and the condenser diaphragm at the front aperture of the condenser.. Each has an optimum setting in a properly adjusted microscope. Above: Microsystems DMI600 B inverted microscope; below: Zeiss Axio Observer inverted microscope." [54] Image adapted from [54].

3.2.3 Digital imaging

Quantification in modern life sciences using a fluorescence microscope is essentially bound to digital imaging, as digital images are the output of microscope cameras. Digitalization (mathematically and technically) is a complex topic in itself and in the following only a brief introduction should give a basic idea how the process of digitalization for imaging platform is employed.

"Charge-coupled device (CCD) and complementary metal oxide semiconductor (CMOS) detectors are small, centimeter-size chips of silicon that are divided up into millions of tiny picture elements capable of storing photoelectrons during an exposure. The photon count in each pixel is then digitalized and displayed on a computer monitor or other display device. The light-sensitivity, dynamic range, and spatial resolution of these detectors are extraordinary. The efficiency of light collection is so great that even weak fluorescent images in a microscope can be recorded in just a few milliseconds. Because they give a linear response over a large range of light intensities, CCD and CMOS cameras can function as imaging spectrophotometers, producing tens of hundreds of times better resolution of light intensity than video or film cameras. They also have a high spatial resolution comparable with film [...] and can acquire "full frame" images at close the standard video rate of ~30 fps. Because digital imaging is so fast, one can see and interact with the images on the computer monitor in real time.

The combination of microscope and digital camera, together with computer imaging software, defines what is called a *digital imaging system* ([**Figure 11**])., a mainstay of the laboratory that has greatly stimulated the use of light microscopy in research [75]. To use the equipment properly, some basic study is needed to master several software-dependent procedures for image acquisition, processing, analysis, and display. [...] the principles and design of the CCD camera [is presented in **Figure 11**]."[54]

Figure 10 depicts a sample result of a multi-color imaging. Each optical channel is read out separately and quantified into a grey scale matrix.

Independently from the specific technology of the camera chip, CCD as well as CMOS and other chip technologies are based on similar principles of photon detection and counting as it is illustrated and explained in **Figure 11**. Relevant parameters of an imager (camera chip) are the following:

- The resolution
- The pixel size
- The bit depth (i.e. 12 bit for 4096 quantification units)
- Dynamic range (range of min to max signal intensity)
- Maximal Frame rate
- The signal to noise ratio
- The quantum efficiency

Depending on the imager, other parameters and special options are available. Often the signal can be amplified and some chips offer fast readouts for a configurable ROI (region of interest).

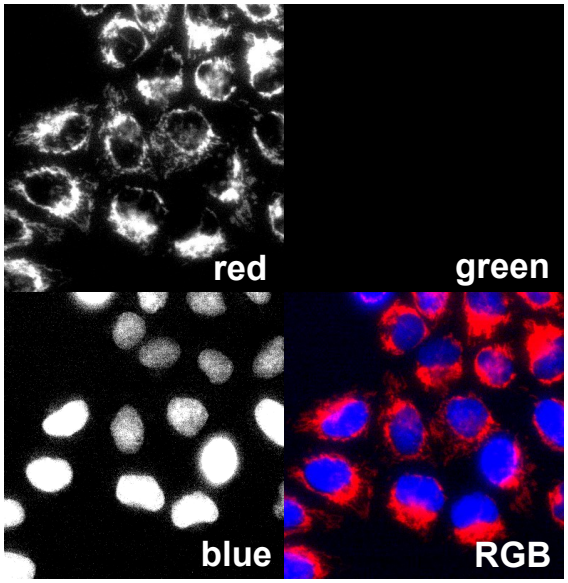


Figure 10 Multi-color imaging result sample. Each optical channel is read out separately and quantified into a grey scale matrix. This can be done sequentially by changing the excitation and emission filters or by splitting the beam into different paths which are imaged by separate cameras. Each channel image is a gray scale image and only for visualization the grey values are translated into color values again.

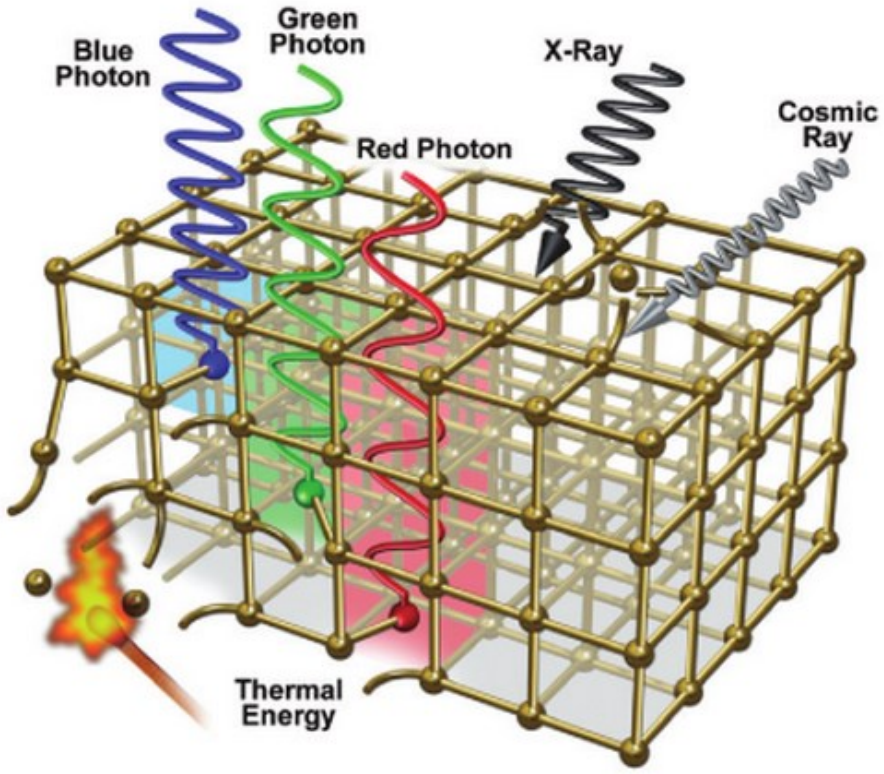


Figure 11 "Silicon as a photon-sensitive substrate in a CCD imager. The sketch shows the effect of incident photons of various wavelengths on the silicon matrix of a CCD. Incident photons interact with the silicon, breaking covalent bonds between the silicon atoms and generating electrons and electron-deficient sites called electron holes. A voltage potential applied across the CCD holds the accumulating photoelectrons in the silicon matrix until they are read off from the chip and digitalized. Red photons penetrate deeper into the matrix than green photons followed by blue photons, accounting for the relative insensitivity of silicon to blue light. High-energy x-rays and cosmic rays disrupt many bonds and generate large saturating signals. Typically, there are a few cosmic ray hits on the CCD surface per minute. Thermal energy, represented by the match, also disrupts bonds and generates electrons (thermal noise) that cannot be distinguished from photoelectron counts; however, the problem can be reduced significantly by cooling the CCD to very low temperatures. After the electron charge packets are read off the CCD surface, the structure of the silicon matrix is restored, and the CCD is ready for another exposure." [54] Image adapted from [54].

3.2.4 Digital image processing

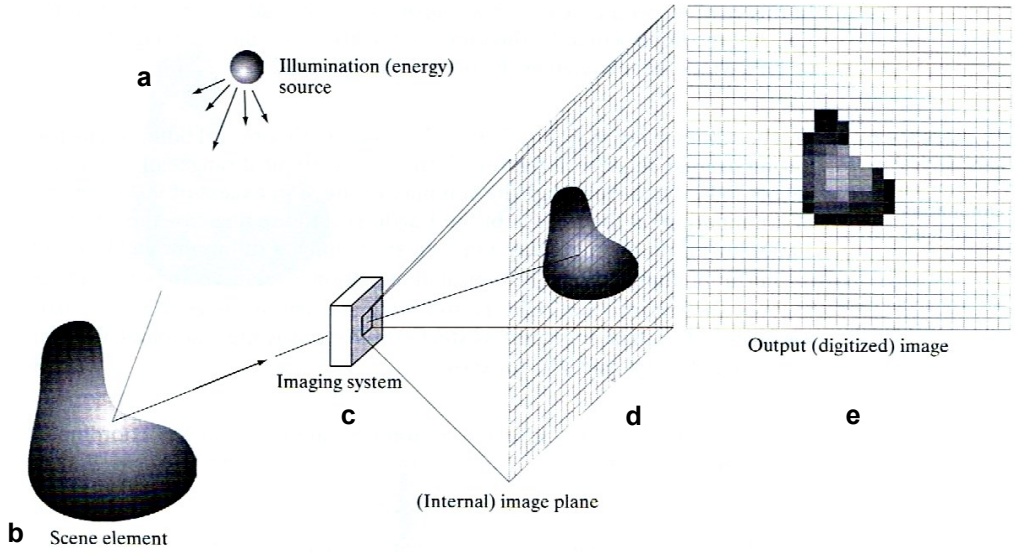


Figure 12 Digital image acquisition process. (**a**) illuminiation source, (**b**) scene element, (**c**) imaging sytem, (**d**) image plane, (**e**) output image. Image adapted from [54].

As illustrated in **Figure 12**, the scene element is digitalized as a matrix of gray values. The process of the digital image acquisition is bound to physical limitations of the imaging system. First of all, the detector of the imaging systems needs to be able to quantify a certain amount of photons which are emitted from the light source or the object itself, i.e. in the case of fluorescence microscopy. The so-called dynamic range of the detector defines the min-, max-boundaries within which the detector is able to quantify light dependably. If the spacial resolution is relevant, some limitations to consider include: the pixel size in comparison to the point spread function of the optical system, and the emitted wave length. The quantification itself is not necessarily linear, which needs to be considered when evaluating gray values of an image. Additionally, quantifying the emitted light over time causes known issues of time discretization. Depending on the sampling rate of the quantification, information of different object features may change due to under-sampling of fast processes. Image processing, however, often is not aware of the physical properties of the imaging system, which can lead to wrong results. It generally is comfortable to directly use the output image without any knowledge how it was generated.

3.2.5 Parallelism

Developing and evaluating the LifeXplorer, fast data processing was the overall goal to be able to provide computational decisions for feedback controlled microscopy as fast as possible. Exploiting concurrency in processing therefore was a main objective. Concurrency can be achieved through different methods and levels. One method is the so called parallelism. It can be described as the following (translated by the author):

"Parallelism is [...] the short form for the possibility to execute operations in parallel". [76]

In fact two types of parallelism are discriminated: data and program parallelism. This differentiation however will not be taken into account, since in the following considerations about parallelism it is assumed that the execution of operations has no data access limitation in order to simplify the complexity of this topic for the reader. On the local computer there are in principle four program levels, and thus four levels of parallelism from the computer architecture point of view.

Table 1 Levels of parallelism. Adapted from [76]

Program levels	**Characterization**	**Degree of parallelism**
User programs (jobs)	Multiple jobs are executed simultaneously	Low
Processes (tasks)	Multiple processes are run concurrently	High
Commands	Multiple commands of one program are run concurrently	Low to high
Elementary operations	An expression consisting of multiple operations is executed.	Low

Table 1 shows all four program levels with their characterization and degree of parallelism. The parallelization degree on the level of user programs is low. The degree of scalability on the same level however is very high, especially when user programs are distributed across the borders of a single processing node on a

processing cluster. Real parallelism can only be reached if there are multiple function units which are able to execute operations simultaneously. In computer architecture and any other production systems two general approaches exist to reach parallelism:

1. Pipelining
2. Parallel function units

The principle of pipelining is to separate sequential processes into autonomous, specialized units that are aligned. A vivid sample for pipelining is a sequential assembly line as it can be seen in **Figure 13**.

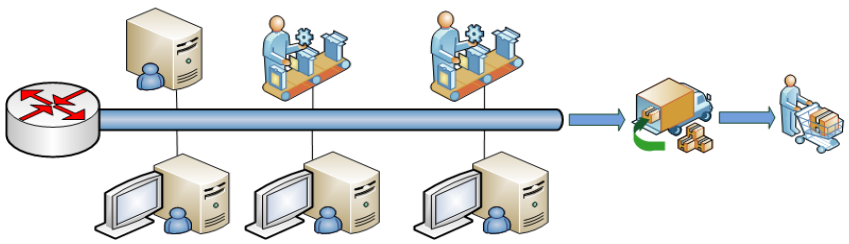


Figure 13 Sequential assembly line

The production process of an assembly line is split into single work steps, which are exactly synchronized. Each working step is connected to another working step by a rigid transfer line [77]. Parallelism in the approach of pipelining can be imaged the easiest if one assumes goods are permanently produced. This causes every work station to be permanently busy producing parts of the final product. The desired concurrency is then reached by executing single operations simultaneously. The second approach to reach parallelism is the more obvious: to use parallel units for the same task. **Figure 14** illustrates the usage of multiple assembly lines to reach a higher degree of parallelism, while every assembly line is parallel itself implementing a pipeline for parallel operations. To have more than one work station for the same task however is only efficient if the degree of parallel stations can be exploited. In both approaches the scheduling and fetching of new tasks has to be managed well to fully load the production lines.

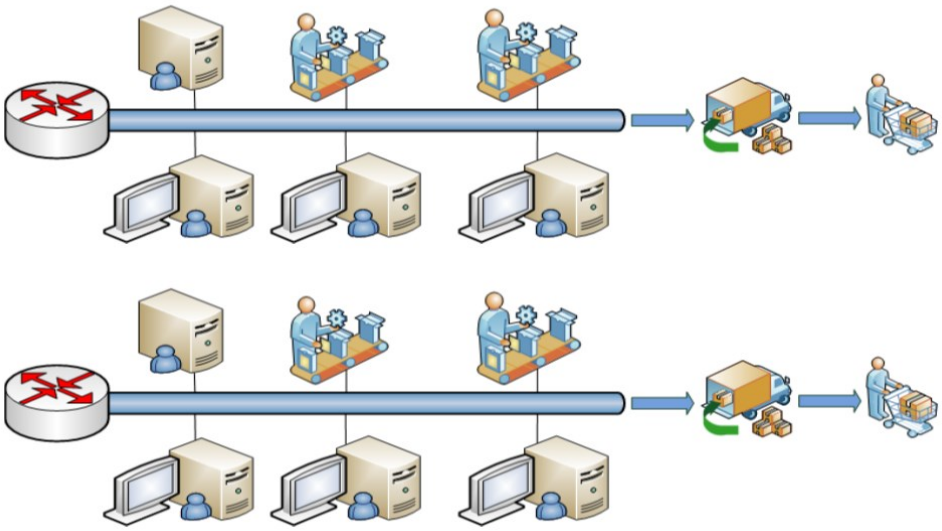


Figure 14 Parallel assembly lines

"The aim of every software system must be to efficiently load all active units in order to increase the throughput of the complete system and thus to be able to execute applications as fast as possible." [78] (translated by the author)

The partition into four levels of parallelism mentioned above can be slightly changed from the software development point of view. From this point of view a different partition into three levels is easier to understand:

- Process level
- Thread level
- Node level

Processes have their own memory area, clearly divided from other processes. The communication between processes as a general rule is only possible with the intervention of the operating system. Threads in contrast can share the same memory area and their synchronization has to be implemented explicitly by the software developer. A process can consist of multiple threads which can run simultaneously depending on the computer architecture. A node in this context is the term for a computer running in a network of other computers. This level of parallelism is in principle process related, however parallel processes on different

nodes can exploit local parallelism on the thread level. All levels of parallelism can be exploited in the LifeXplorer.

3.2.6 Distributed Systems

"A distributed system is one in which components of connected computers communicate and coordinate their actions only by sending messages to each other. This definition leads to the following characteristics of distributed systems: the concurrency of components, the absence of a global cycle, and the independent breakdown of single components."[79] (translated by the author)

The most famous example of a distributed system is the internet, with both its soft- and hardware distribution. Systematically the potential of this distribution becomes obvious and is communicated in modern terms like "cloud" and "cloud computing". Although these terms do not directly refer to applications running on the internet, more and more applications "in the cloud" do so. High performance and supercomputing [80], which refers to the term super computer, brings highly parallel computers, mostly with specialized computer architectures, into application. For science related applications, it has a high potential to be a solution for major computing issues in a large variety of applications [81]. Medical [82-84] and biomedical imaging [85], applications of molecular biology [86, 87], cancer research [88] as well as drug discovery [89], car simulation and crash test simulation, astrophysics, high energy physics experiments, optics [90], fluid dynamics [91-93] and modern weather prediction [94] depend on the existence of distributed and highly parallel systems to compute solutions of given computing problems. Life sciences to a large extent start benefiting from existing approaches to exploit distributed systems, such as the three-dimensional electron microscopy reconstructions [95] and other complex computing issues, such as large scale image screen. For example, genome wide RNAi screens [96-100] could not efficiently be evaluated without the usage of parallel computing and thus distributed systems. Another application is to efficiently search sequences as a challenging task for bioinformatics [101], which becomes even more relevant since

the genome sequencing is pushed by the International Cancer Genome Project (ICGC) where 50 different cancer types are sequenced across the globe [102].

The basic principle for distributed computing and applications is the modularization of programs in several parts that are independent from each other. Hardware parallelism can then be exploited by distributing computing tasks to different computing units and by running several tasks in parallel. Essentially communication and synchronization with this basic step comes into play. If an algorithm is separated into different independent parts and run on different hardware units, these units have to communicate their intermediate results in order to compute the total result of the whole algorithm and for all existing data packages. In principle a distributed system and application can be run locally on a single computer as well in many cases. Scalable applications are designed to be distributed on different computers and in the best case in a way that the application scales automatically when adding computers to the processing network. The fixed costs to develop scalable applications are in general higher than the costs of single node applications, which are not able to make use of multiple computing resources. The need to build distributed applications increases, because computing problems are becoming more complex. At the same time, the application design itself becomes more complex, because non-expert users also want to benefit from a combined set of technologies.

Because the distribution of computing tasks implies communication, the latency and bandwidth of network connections, or rather the overall system, become crucial parameters to run high performance computing applications. As soon as the communication overhead becomes higher than the increased computing performance by the distribution of tasks, the overall performance of the distributed system will be lower. For the development of distributed systems and applications a large variety of architectures and approaches exist. These approaches will not be further discussed in this thesis, however main principles of the communication will be explained in the following, in order to provide a basic background the final architecture of the LifeXplorer framework. Communication in distributed systems can be divided into two classes: synchronous and asynchronous.

Synchronous distributed applications

Applications which are synchronously distributed block the local program execution until a network message of a communication partner is received [79].The result can be that the local process can freeze if no time out logic is implemented or an existing time out can cause the program to fail if the result of the communication partner is necessary for a successful execution of the local program code. The parallelism of the computing doesn't have the highest priority in such applications. Although the work load of the local resources is reduced if a communication partner takes over calculations, local resources are not actively used for the same application. Nevertheless synchronously distributed applications can benefit from heterogeneous resources, which are used on demand for special purposes. A microscope host computer i.e. might not have as high a performance as a different computing node of a computing cluster. As soon as the communication overhead is lower than the amount of time that can be saved by computing a task on a distributed computer, the synchronous and distributed task execution can already be an added value for the application performance. This basic communication pattern of distributed applications is called Client-Server architecture [79]. The server in this case provides services to different clients, which in general have a less strong hardware. A typical example is a database management system. If a database file is shared by many clients on a network drive, the performance is in general much lower than it would be using a database management system. Data base management systems offer services to read and write data to the database locally and in an ordered an optimized way. In this case only relevant data is sent over the network, and in addition the server can have an optimized hardware, whereas the clients can have weaker hardware.

Form the process point of view, synchronous communication mainly offers one advantage: no synchronization if necessary. The process itself therefore becomes easier to understand and less error-prone. In addition, the process can be understood and recovered more easily on runtime. Developing synchronous, distributed applications is closer to the development of non-distributed applications than it is the case for asynchronously distributed applications.

Locally blocking the program executing in any case avoids using resources in an optimized way. Exploiting existing resources however often leads to asynchronously distributed applications. Parallelism instead of pipelining moreover can optimize both latency and throughput.

Asynchronous distributed applications

Distributed applications which are run asynchronously do not block the program execution when communication with another distributed processes is started, which means that the local program continues to run while in parallel the communication partner receives and handles a message and asynchronously provides a reply. Depending on the technology, such as MPI (Message Passing Interface), it is possible to retrieve messages via a communication status register. This means that received messages can cause a soft- or even hardware trigger, which pauses the current program execution and can then be handled by a special synchronization code that actively handles the message of the communication partner and re-schedules the result into the local program execution. Asynchronous applications can be understood as a super class of synchronous applications, simply because synchronicity can be simulated with an asynchronous application. On a lower level, every network communication is normally asynchronous, because the operating system can and will continue to run after a message is passed over to the network interface. As soon as a message comes back, the local program execution will be continued based on triggering or polling. In both cases however, the underlying network communication itself is normally asynchronously implemented to ensure that multitasking is still possible. Once an asynchronous application is implemented in a way that looks like a synchronous one to the operator, the underlying benefits of asynchronicity cannot be exploited anymore. Alternatively it is possible to locally simulate asynchronicity by creating threads for each asynchronous program call. Internally the communication then can be asynchronous, while the local program execution is still asynchronous. In the case of the LifeXplorer asynchronicity plays a crucial role for the implementation and understanding of the distributed system. Because hardware and computing tasks can be run asynchronously, the re-scheduling of

messages is a relevant design challenge. Asynchronous software designs are often based on threading. The sending process itself can be encapsulated into a thread and only the send communication is called by a different thread that awakens a new thread, which runs the send logic. The same pattern can be used for the asynchronous receiving process. A hardware or software trigger (which can be implemented by so called polling as well, which in general implies high resource usage) makes the operation system change the program execution by jumping to a program code position that handles the message. Like in a hardware system, the software design is then based on encapsulated sending, receiving, and other logical units that are run in parallel. A relevant question is how exactly the binding between the communication thread and the program logic is implemented. In order to show the complexity of this topic, one example of highly scalable software designs should be mentioned. The so-called instance per call mechanism creates a new thread for each communication call. This pattern is highly scalable because memory is freed once a call is terminated, and at the same time, a hardware with several CPUs can handle several communication calls in parallel, which can highly increase the overall availability and throughout of distributed applications, especially in so called SOAs (Service Oriented Architecture). At the same time, the overhead is a lot higher for handling as many communication threads as necessary in parallel, in contrast to only one. The overall performance therefore is initially reduced by this overhead, the system is on average slower, but scalable. Asynchronous systems, as briefly explained, are a lot more complex as they demand proper designs, make debugging more difficult, and offer a large variety of different implementations on how to make use of asynchronicity. Unfortunately, asynchronicity offers the beauty of exploiting resources more efficiently and once a working design is successfully implemented it is often more robust due to the modular design principles asynchronicity demands. In principle, the design and maintenance of such applications quickly gets more complex. This is mainly due to the assignment of the data of the asynchronous program flow, which may prove to be much more difficult. While a synchronously implemented program by its flow design has the right data in place, when needed, an asynchronous program can have a completely different program state at the time a reply message is received.

The original program status that was valid before a message was sent may be reassigned by the usage of a status history. This issue in computer engineering is called context switching and takes place in the CPU registers every time a different program is run on the CPU. While CPU and operating system designs offer this ability by default to make multitasking possible, software applications have to implement such a behavior explicitly depending on the concrete needs and underlying technologies available.

A network of hardware components which interconnect with each other always is the physical basis for a distributed system. The LifeXplorer system was designed to connect heterogeneous computers with each other, using their local computing resources such as CPUs and GPUs (graphical processing unit). Since heterogeneous networks can consists of different hardware and software platforms, the challenge of building the LifeXplorer was to build an interoperable communication system that abstracts from the specific hardware and operating systems. In the same time, the communication should be based on modern object oriented programming approaches to be able to build robust and scalable software architecture. The solution for the LifeXplorer development was to build it on the so-called Service Oriented Architecture (SOA) [103] paradigm. This architectural paradigm helps to build robust distributed systems and applications and is further explained in the following chapter.

3.2.7 Service Oriented Architecture (SOA)

Distributed applications are complex, as becomes obvious from the above. Designing and maintaining them is a challenge. In order to handle the complexity of software applications, it is often helpful and even necessary to make use of so-called design patterns [104]. They provide general approaches to solve specific design problems and help make it easier for the developer to create a clear understanding of the solution being designed. This furthermore helps to develop robust applications more efficiently. Especially the so-called lose binding plays a central role in modern applications. Loose binding basically means that

components communicate only through small interfaces, whereas local changes will not affect other components. One approach for this strategy, which respects the cooperation of multiple, distributed function units, is the so-called service oriented architecture (SOA).

"A service is a functional unit, which is brought into contact with the world. In this sense, it is the next evolutionary step in the long journey from functions to objects to components to services. The service orientation (SO) is itself an abstract figure consisting of approaches and good experiences to develop service-oriented applications." [105] (translated by the author)

During the history of software development, new methodologies and technologies have always taken advantage of established ones. However, as mentioned previously, new challenges simultaneously come along. Generally, service oriented architectures are based on a standardized object description and communication protocol called "simple object access protocol" (SOAP). On this basis it is possible to transfer objects across heterogeneous systems and to execute method calls of objects that are not locally implemented (**Figure 15b**). This is done by the so-called serialization of the object at the transmitter and the de-serializing on the receiver side. The receiver only needs the structural information of the received object that is present based on XML, which can be integrated even on runtime. The service provides operations (**Figure 15a)** and data to other clients, which can consume the service through standardized communication channels (**Figure 15b**).

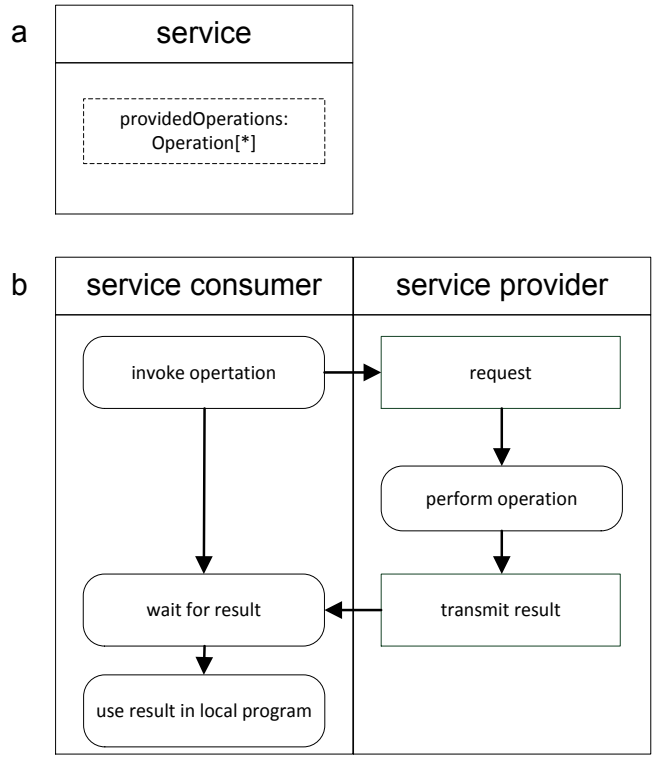


Figure 15 Basic service architecture. (**a**) A service offers operations to other services. (**b**) Service consumes other service to work with its computing result locally. Images adapted from [106]

The approach is close to real life, where organizations and individuals offer services for others. On runtime services can be called and consumed. The underlying program code can be exchanged even on runtime as long the interface stays the same. Although complexity is always higher than the ability to manage complexity by more sophisticated approaches, SOAs seems to have made a solid step towards future applications, which will be based on distributed systems and services. Every intelligent network consisting of heterogeneous participants that communicate with each other can in principle be designed as an SOA. This exercise almost always has the desire to develop separate functional units as cleanly as possible, and to be able to reuse them, which only can easier be done by thinking in the design pattern and paradigms of an SOA. The real gain, however, comes through the standardization of this approach. An SOA will show its beauty only by the fact that developers focus entirely on the design and development of robust features, without having to lose too much time on the implementation of the communication. The merging of the separated functional units is facilitated considerably.

3.2.8 Organic Computing

Self-organization is an attractive design principle for a large variety of applications, where system parameters change over time and re-adjustment and autonomic optimization of parameters is possible. Organic computing tries to apply self-organization in biological systems to information systems and is thereby closely related to systems biology [107]. It can be considered an evolutionary system optimization in dynamic environments [108]. The quality of the self-organization is describes in self-x features. As organic computing approaches are fundamentally based on the structure and thus architecture of a system, in the following basic architectural design templates are briefly described. Eventually organic computing tries to provide generic architectural templates on which system designers can build more flexible systems, which are at the same time more robust against crashes. Like in nature, applications run on organic computing architectures are process driven. Structures and topologies may change over time due to runtime decision of the organic computing management units, which are often called observer and controller (**Figure 16**). The productive system is observed on runtime by the observer unit, which reports the results of the observation to the controller. The controller is configured based on pre-defined goals, such as availability-, throughput-, and latency-optimization. Depending on the reported system status the controller can change the productive system, its parameters, and even structure. The productive system is abstracted in a process definition rather than a structure, which drives a certain process. In this way, the process which should be driven is dominant, over and against the structure, while the controller tries to drive this process, building or changing structures on runtime if necessary. Organic computing systems by design are modular and often distributed intelligent systems and can be considered autonomous computing networks [109]. As a practical example of a process, we can image a data analysis workflow. It can be run on a computing grid, with fixed associations. I.e.: algorithm 1 is run on node 1, its results are sent to node 2 where algorithm 2 processes the input of algorithm 1 and creates the final result. Organic computing in contrast will change the assignments and

communication pattern autonomously, i.e. if node 2 crashes another node will be integrated into the productive system structure in order to run algorithm 2. Another case would be that on runtime algorithm 3 and 4 are added and another communication structure is necessary. Organic computing thus reduces complexity for the operator and increases the reliability of a system.

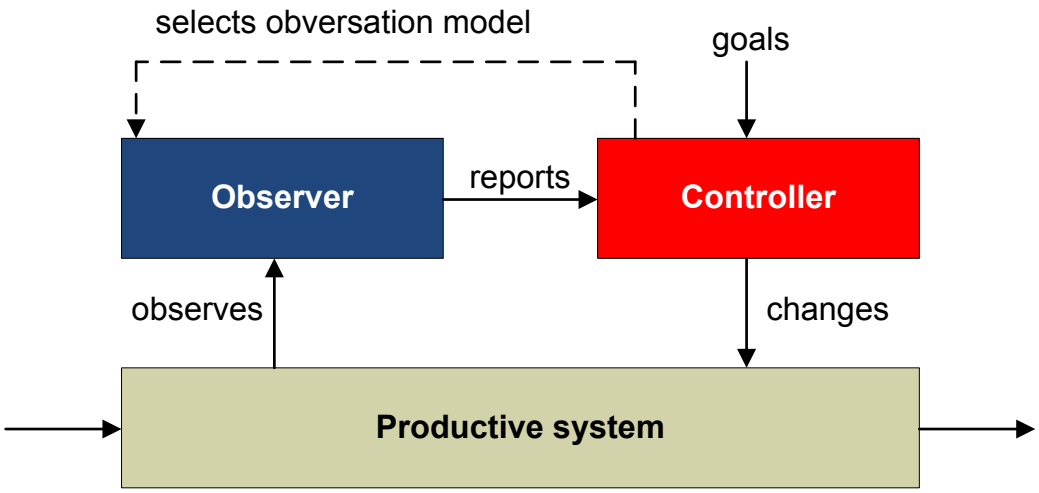


Figure 16 Basic observer / controller architecture. Image adapted from [42].

As in biological systems, organic computing systems are designed to react on changes rather than assuming that the overall system behavior is predictable, and fixed structures therefore can be optimized before runtime. Especially in computing, the applications and algorithms choose different paths in their sub programs depending on the data and the system history. This application path however cannot be predicted in many cases and only on runtime, when processing application-, data- and time-specific information, different needs for the system configuration appear. Organic computing is closely linked to neural networks [110] as they implement similar principles of self-organization and flexibility. They can be integrated into organic computing strategies as machine learning strategies. Neural networks can be trained during runtime and dynamically adapt to the environment, driving a defined output function.

To measure the degree of autonomy, the following parameters of systems have to be considered. The complexity reduction R of self-organized systems is described as:

$$R = V_P - V_{HL} \quad \text{(Eq-7)}$$

whereas V_{HL} is variability the high-level configuration space (i.e. the goals configuration of controller) and V_P the variability of the internal configuration space (i.e. the productive system) [42].

The degree of autonomy S is defined as:

$$S = \frac{R}{V_P} \quad \text{(Eq-8)}$$

If no external control ($V_{HP} = 0$) is necessary, the highest autonomy is reached for and $S = 1$. As organic computing systems are designed for dynamic environments [108], the autonomy is not just a static value, but can change over time. The dynamic complexity reduction within the time window $t_2 - t_1$ therefore has to be defined as:

$$r = \int_{t_1}^{t_2} (l - h)dt \quad \text{(Eq-9)}$$

$$s = \frac{\int_{t_1}^{t_2} (l - h)dt}{\int_{t_1}^{t_2} (l)dt} \quad \text{(Eq-10)}$$

whereas $h(t)$is the high-level and $l(t)$ the low-level flow of control information [42]. As illustrated in **Figure 17** the degree of self-organization increases with the variability of the productive system. Three basic design patterns for organic computing architectures are shown in (**a**)-(**c**). (**a**) depicts the central observer-

controller architecture. For the whole productive system, only a single observer-controller instance is reliable. A multi-level pattern is shown in (**b**). The productive system itself is segmented and each segment has its own self-organization, whereas the overall system is controlled by an observer and controller. The highest autonomy is given if the segmentation rate of the productive system is maximized, so that the variability is maximal and each segment has its own self-organization, without any additional control unit, as it is illustrated in (**c**). Practically the fully distributed self-organization of (**c**) however leads to high redundancy. Each sub system will have to measure its environment and no specialization allows creating and sharing common knowledge about the overall system or sub systems of it.

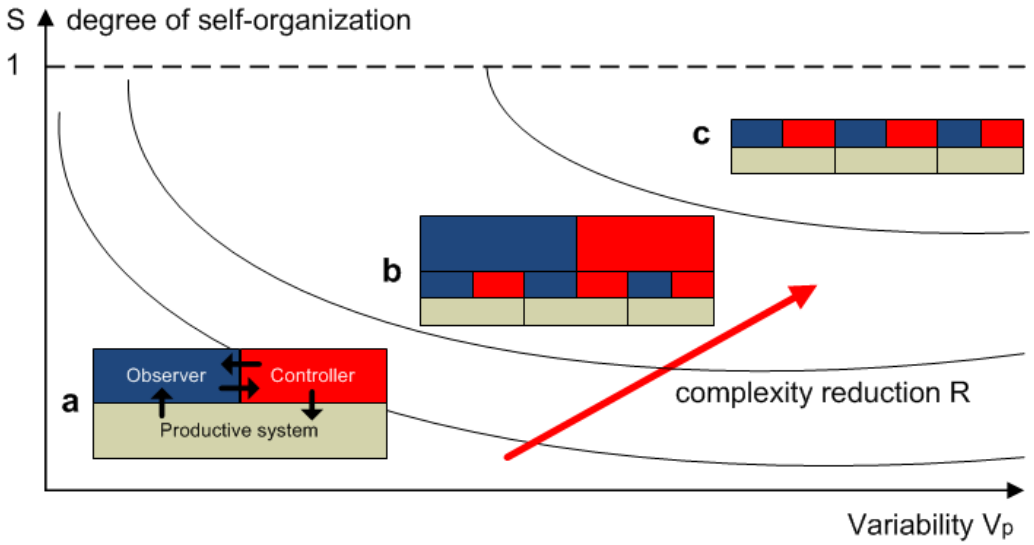


Figure 17 Degree of self-organization S as a function of the variability of the productive system, V_P

In LifeXplorer the central approach of **Figure 17** (**a**) was chosen, as it is the easiest to trace during runtime. The multi-level architecture offers, however, more flexibility and reliability for the application run on such architecture. Having only a single instance for self-organization, already the measurements of the observer can create a bottleneck for the communication channel of the node which processes the messages of the observer. In addition, redundancy of the

observer and controller instances is an issue as these components are highly critical system components, needing to be stable against environmental changes and crashes. This leads to a hierarchical topology and includes strategies for redundancy, which are not explicitly explained here. Once it is modularized, the productive system can however be adapted dynamically such that it is stable against crashes. Having several processing nodes in a network, the controller can switch tasks from one node to another. Having memory afflicted tasks, demands to implement synchronization strategies, which are also not mentioned in more detail in this thesis. Redundancy and multi-level strategies have been investigated, but were less relevant than the Automated Cost / Result Optimization during runtime.

3.2.9 Real time

Since the ARCO algorithm necessarily needs a feedback control system, real-time aspects had to be considered in order to implement reliable parameters optimization methods on runtime. If the duration of a processing operation is reliably fixed, a system can be called a real time system. If an expected time for a processing operation is smaller or equal to the real-time interval, a system can be called a soft real-time system. Hard real-time implementation is only necessary in systems which dependably have to guarantee that the real-time requirements can be strictly kept for all tasks, meaning that tasks or signals are not processed faster than expected. However in general this creates hard limitations to the system complexity and therefore is often only possible and necessary for low-level implementation in communication and electronics. On the software level, soft real-time commonly is suitable if the architecture implements buffers to catch signals and task into a list first, to be stable against variable processing times. Practically, real-time played a role for the LifeXplorer platform, because one cycle is determined by the physical imaging workflow and can artificially be extended by the operator if the imaging software allows such a configuration to extend the cycle time by adding adaptive pauses after the imaging workflow. Finally the processing time of data analysis workflows determines the minimal

imaging workflow time which is necessary to successfully couple together imaging and data analysis in real-time, in order to control the imaging workflow on runtime. LifeXplorer was not explicitly designed to support soft real-time for data processing, however it can synchronize the imaging workflow with the data processing and thus clock the imaging cycle with the data processing. In the worst case, the sampling rate of the imaging workflow therefore will be extended artificially and cannot reach the actual physical limit.

3.2.10 Software platform .NET

.NET is a runtime based, cross-language, object-oriented software development framework. The primary aim of such a development framework is to reduce complexity of software development by providing comprehensive and standardized libraries, which can easy be used and combined within the customized application development. Among other things, this aim is achieved by abstracting from the underlying operating system. Source code that is writing on .NET can be used on every operating system for which the .NET framework is available. The developer does not need to know the specifics of different operating systems. To make this possible, developers can write their applications in different languages, such as Visual Basic, C#, C++, PHP, and many others, which are internally translated into the so called Common Intermediate Language (CLI) [111]. The source code of an application is then translated into machine code on runtime by the Common Language Runtime (CLR) (**Figure 18**).

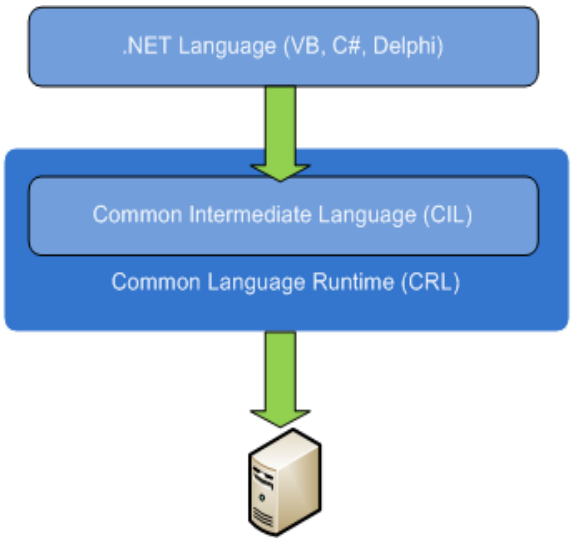


Figure 18 .NET framework - language translation

This basic "compile on demand" approach, which is implemented by every interpreter of a scripting language, creates a relevant overhead, which decreases the performance of applications. Fortunately .NET in contrast uses more sophisticated approaches and incrementally precompiles source code on demand. This approach then leads to lowered performance loss. .NET however can be considered to have a lower performance than for example C++. The main reason for this is that .NET does not by default allow the use of direct object access through so called pointers. They give the developer the chance to work more or less with the physical address of an object in memory. Because this has been a major security and stability issue of existing applications, .NET managed the access and developers can only access an object through the .NET layer. The overhead therefore can be understood as a function call. Every time an object such as a variable is accessed, a function of the .NET framework is called in order to make sure that no access violation happens and if that is not the case the content of the object is returned. The term for this internal behavior is called managed code. Like the Java Virtual Machine (JVM), .NET is active and manages the program code even on runtime in the case of the already compiled machine code. Because direct object access is not possible, memory and memory

addresses can be managed by .NET on runtime. C++ applications and others, where direct object access is possible, often are not dependable due to memory management related issues. .NET applications are less complex for developers in this case. They can free objects and trust the framework to handle the underlying memory management with a high dependability. The robustness of applications can therefore be increased only by using frameworks such as .NET, because the so called garbage collector (GC) is a highly developed memory management core unit, addressing memory leaks and optimization issues. In practice, however, developers have to be aware of the main principles of the garbage collector as well, and have to have knowledge how to make the garbage collector work properly.

The fully object oriented approach of .NET is a basis for proper and robust software development. It makes applications in principle more scalable and less error-prone. A unique feature of .NET is the already mentioned cross-language integration. Almost every high language such as C++, Java, or Delphi can be used. The added value of these languages for the developer is that almost every standard library of the framework can be used and integrated easily. This in general makes software development more elegant, efficient and effective. The C# language has the highest compatibility with the .NET framework and all its libraries. It was designed and optimized to be a pure .NET programming language. The LifeXplorer and all its components except hardware drivers were implemented using C#. In many various areas .NET is furthermore strongly XML oriented, because XML is used as a standard intermediate format in the case of data communication. This standardization could profitably be used in different components of the LifeXplorer. From the software architectural point of view, frameworks such as .NET add value because they serve as a relevant abstraction layer, as mentioned above. Although performance is lowered by this additional software layer, the efficiency and dependability of software development is often decisive. As the measurements in this thesis will prove, the overhead of using .NET technologies is not relevant in the case of feedback controlled microscopy applications. C++ was used to implement hardware

drivers, but mainly because the existing libraries of the hardware components are only available for C++.

Software development frameworks like .NET offer new opportunities of generic programming, which are becoming increasingly important. Generic means that components can be added and modified on runtime. At the same time, the main goal of building the LifeXplorer was to have a framework for science automation and especially intelligent microscopy, where data acquisition and analysis can be modified on runtime. .NET therefore was chosen to build a base technology for the LifeXplorer. Because the LifeXplorer is a package of hardware control components, distributed workflow computing and visual programming, also the possibility of efficiently building graphical user interfaces had to be taken into account. .NET in this case offers a novel framework called Windows Presentation Foundation (WPF), where a user interface is fully described in the XML format and rendered into a concrete form on runtime. This abstraction allows building web and desktop applications on the same bases. To build the LifeXplorer, .NET version 3.5 and later version 4.0 was used. In Figure 19 the .NET framework version stack is illustrated. Fortunately starting with version 3.0, the technology development focused on distributed and parallel applications. Out of the .NET technology bundle, the following key technologies were used to build the LifeXplorer:

- Windows Communication Foundation (WCF) – communication framework
- Windows Presentation Foundation (WPF) – visualization framework
- Parallel Language Integrated Query (PLINQ) – can be considered as being an integrated SQL which is optimized for multi core usage.
- Task Parallel Library (TPL) – local computing task parallelization framework

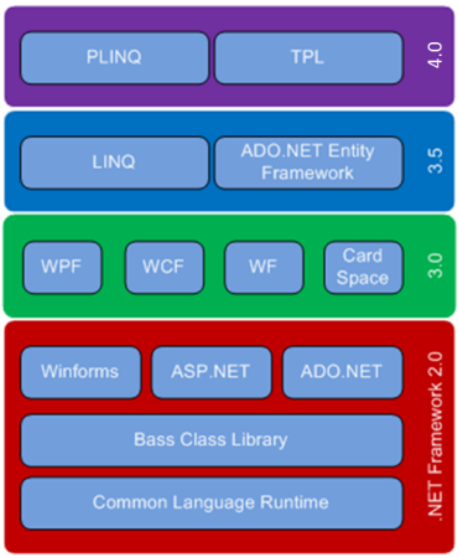


Figure 19 .NET framework version stack

Most of these technologies are only available using the original .NET framework which is built for Microsoft Windows operating systems. However there is a serious initiative called the Mono project that ports .NET to the Unix world in order to make .NET available for unix derivatives like Linux, SunOS, OpenSolaris, OpenBSD, MacOS, and others. Although this project is still at its beginning, a variety of .NET technologies can already be fully used. Mono in addition seems to be a platform with very high performance, as references available on the web show. The LifeXplorer is designed to be a computing cloud that can also consist of personal computers as computing resources, and designed to be available for as many operating systems as possible. Throughout this thesis only Windows platforms were used as the underlying operating system for .NET. Migrating the code to Mono was tested in principle, and proved partially successful.

3.3 Software Architecture

The software architecture of the LifeXplorer platform is based on the ideas of organic computing as explained above. As illustrated in **Figure 20**, the platform consists of two main components. The microscope management system (MMS) serves as a hardware control and data acquisition framework for microscope systems, where preconfigured acquisition workflows can also be set up in a graphical user interface. The MMS is connected to the second component: the processing framework. Both components are connected using services implemented in the Windows Communication Foundation (WCF) WCF implements a framework for service-oriented architectures. During runtime the microscope management systems sends the acquired data to the processing framework, which drives a preconfigured data analysis and feedback control workflow. "Drives" means that the framework is able to permanently reorganize the existing local processing topology, based on organic computing approaches.

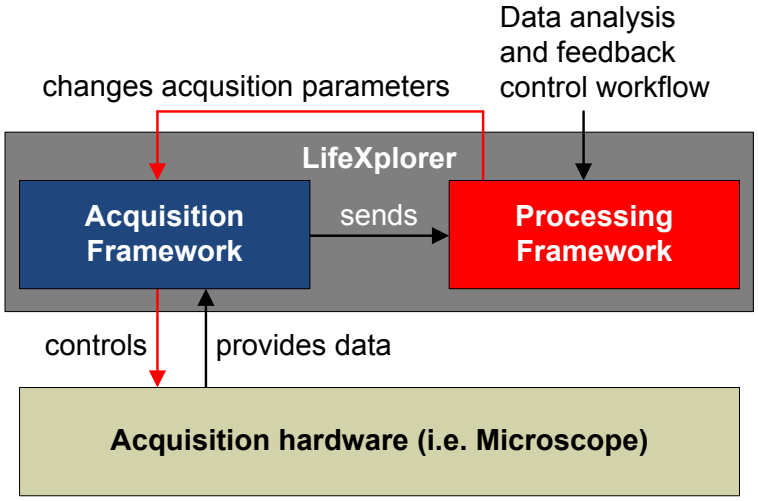


Figure 20 LifeXplorer software architecture.

The processing framework manages a workflow for feedback control. This workflow will process the received data, analyze it, and make dynamic decisions on the microscope system parameters as well as on the acquisition workflow. All parameters that can be controlled by the MMS can also be controlled by the

processing framework. As the framework is not directly connected to the data acquisition hardware, it can generate and send tasks to the acquisition framework in order to change the data acquisition workflow during runtime.

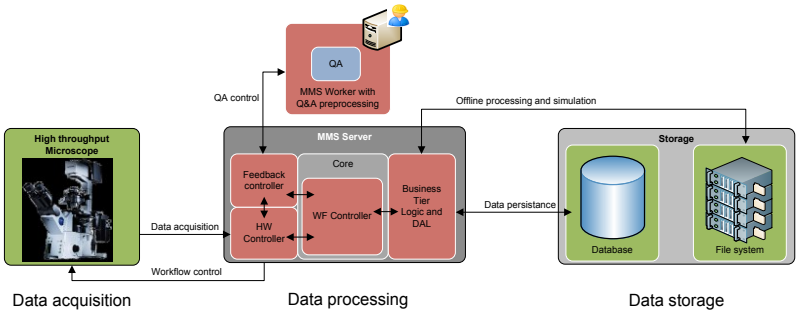


Figure 21 LifeXplorer: first architectural approach with data quality control on an external node

Originally, the aim of this thesis was to improve data quality by feedback controlled microscopy. Because the microscope's host PC has only a limited amount of computing resources, the first architecture for the feedback control system was to push quality control measurements to an external computing node as illustrated in **Figure 21**. In order to be able to understand run time behaviors of the system better, offline processing and a simulation of the acquisition process should be implemented as well. Throughout the project, the architecture was extended due to an additional challenge: making the feedback control system scalable. This was because the hardware system was accelerated to be 10x faster than a standard high-throughput microscope.

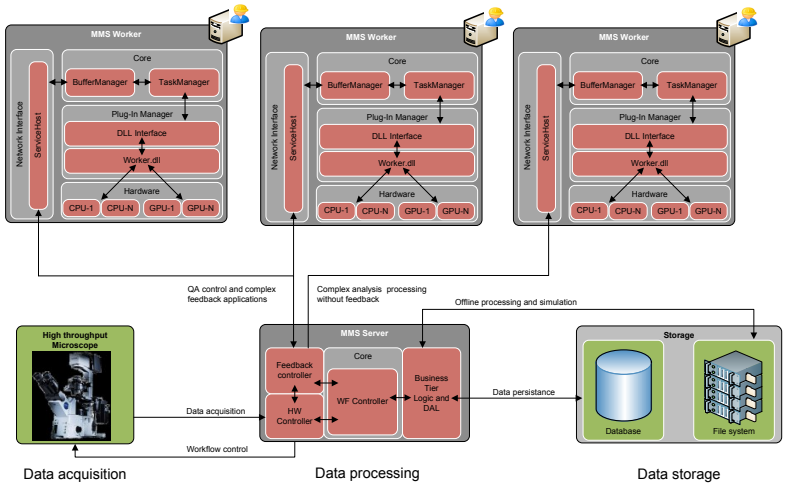


Figure 22 LifeXplorer: first architectural approach with distributed computing on workers.

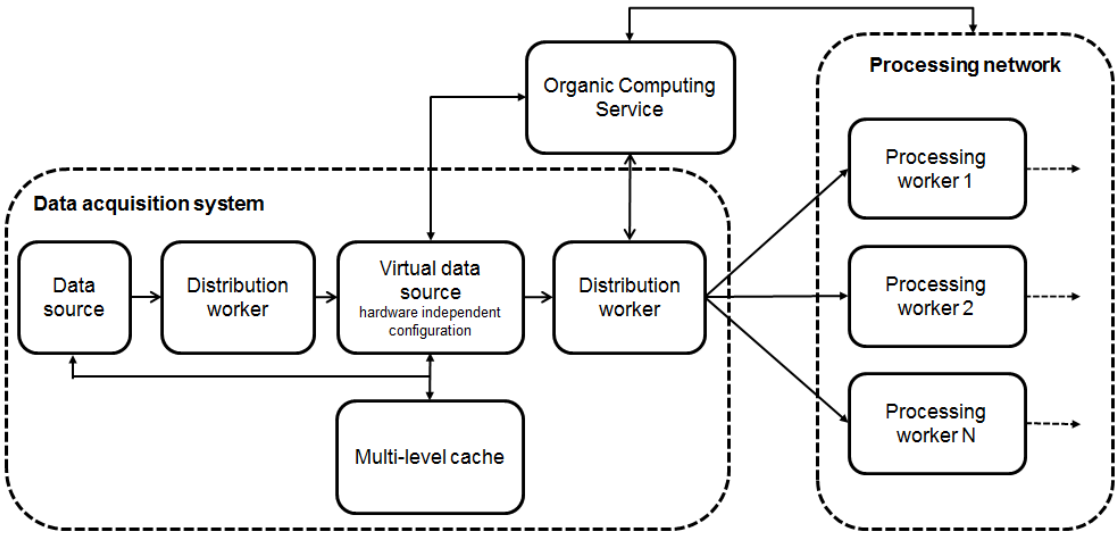


Figure 23 LifeXplorer: data acquisition and processing architecture

The data acquistion and processing architecture of LifeXplorer is illustrated in **Figure 23**. The data source (microscope) provides its generated data to a worker which distributes it to the virtual data source. The reason for this design is that the virtual data source module in general can be instanced on multiple computing notes to balance the work load. After each virtual data source another distribution worker offers the data distribution to the processing network. In reality the virtual data source itself is able to take over the distribution, just like every worker can. In **Figure 23** the design pattern is explained by explicitly illustrating all necessary functions of the architecture. Every function can however be integrated in a single worker or distributed to multiple workers. Another worker function that has to be mentioned here is the organic computing service. This service as well can be executed by all workers. Yet the implemented architecture in LifeXplorer currently allows only one unique worker to take over the organic computing service during runtime. The processing node of this worker also cannot be changed during runtime. This design certainly limits the scalability of LifeXplorer, but builts a first step towards higher degrees of scalability. The existing implementation already allows to bind all workers except the virutal data source to different nodes during runtime. The virtual data source in addition can be flexibly bound to a computing resource by configuration, before runtime. The multi-level cache module serves as a database for historical information. Workers by that can request historical data stacks.

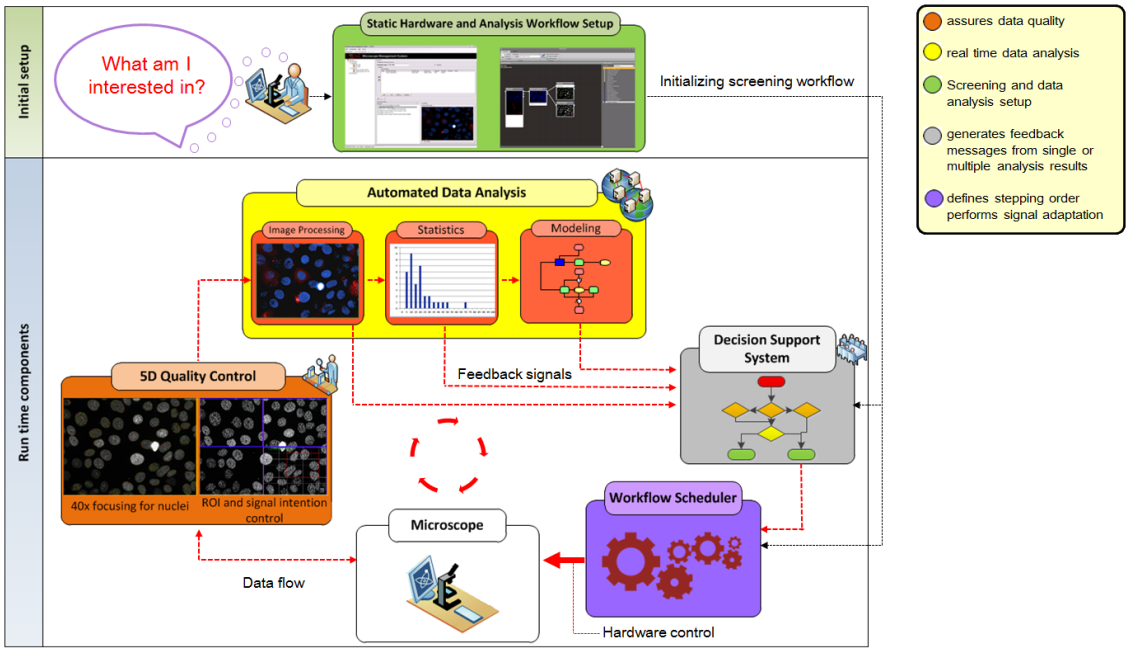


Figure 24 LifeXplorer main components and architecture.

As presented in **Figure 24**, the LifeXplorer platform is built "around" the acquisition hardware: it receives data from it and controls it. Initially the operator has to ask a specific biological question by defining a static hardware workflow and a data analysis answering the specified question. The initial setup more generally implements existing knowledge to automate image acquisition with the aim to improve data quality and information density. The existing knowledge about the biological system as well as its analysis and the experiment definition can be implemented in the MMS and processing framework user interface, the LifeXplorer Intelligence Manager. During runtime four main application scopes of feedback controlled microscopy can be implemented into LifeXplorer. The first scope is the quality control of the data. A typical example in this context is an autofocus function, which also will be evaluated in the chapter Applications of the ARCO algorithm. The second scope can include the whole pipeline of data analysis, starting from image processing, to statistics, and finally ending with the modeling of the acquired data. In each of these analysis steps, processing modules can generate feedback to the microscope in order to make a decision on how to optimize parameters of the running experiment, aiming to increase data quality and information density. While collecting these messages a decision support-system system is necessary to manage different parameter optimizations from different processing steps at the same time. Once a decision how to optimize the workflow is taken, the workflow scheduler scope is the final application scope, where the actual hardware control is implemented and imaging tasks are collected and scheduled. By way of these applications scopes, the LifeXplorer automatically increases data quality and information content.

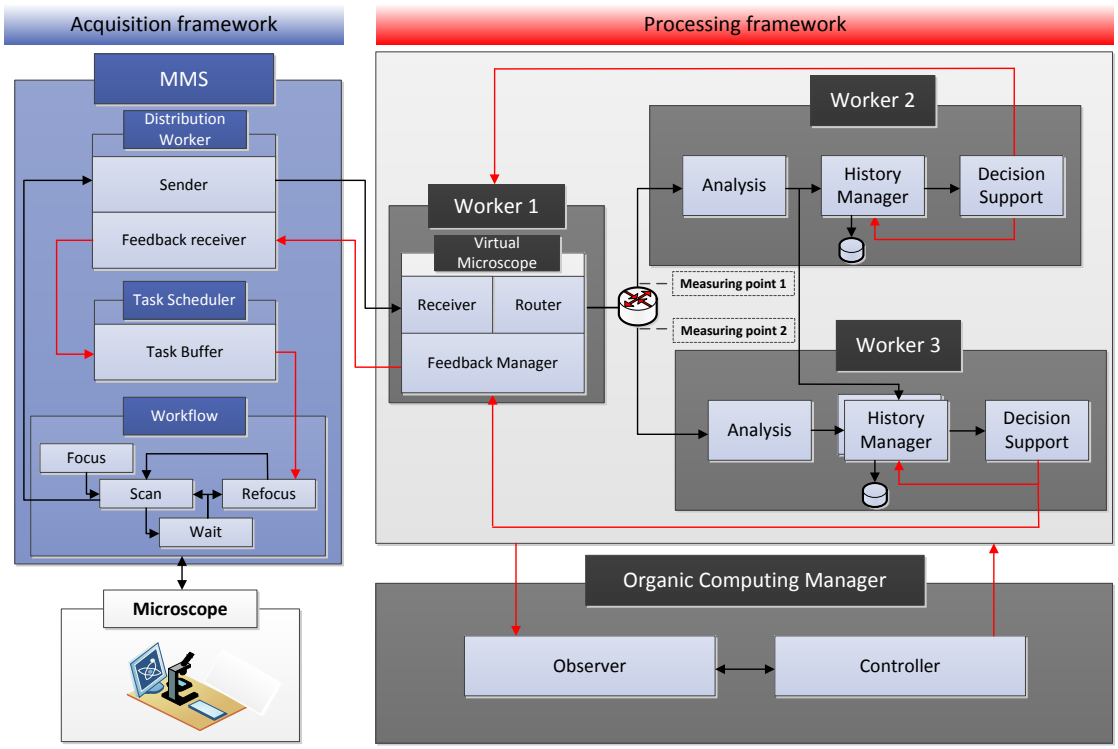


Figure 25 LifeXplorer implementation and application sample of a refocusing trigger.

In more detail, the LifeXplorer main components consist of the sub systems illustrated in **Figure 25**. The software instances virtualizing computing nodes are called "worker". All tasks for hardware control, data acquisition, and processing are run on these workers. Whereas the MMS has a locally fixed setup of software components to run microscope experiments, the processing framework can push tasks to different nodes with different distribution strategies. The physical processing network by that approach is virtualized and the organic computing manager can take decisions on the logical topology based on observed performance measurements of the network. The organic computing controller thus organizes the data processing based on different strategies. One example would be the high availability mode, which is enabled by implementing different forms of redundancy (hot standby and parallel, redundant computing). **Figure 25** depicts the hot standby redundancy strategy. Worker2 is permanently synchronized with historical processing results, in order to be able to take over the processing of Worker1 if the underlying node crashes or is too busy with other tasks to fulfill real time requirements. Once a decision is made on a certain

worker to change parameters of the acquisition workflow, a feedback signal will be sent to the virtual microscope module. This module will normally be run on a worker that is located on the microscope host PC, and virtualizes the microscope hardware for the processing framework. In the case of parallel, redundant processing i.e. it will collect and consolidate feedback signals such that the MMS only retrieves new tasks once. All hardware tasks for the acquisition process are finally managed by the task scheduler of the MMS. Depending on the task, different scheduling strategies can be configured. I.e. in the case of a refocusing trigger task, the scheduler can be configured to execute the task out of order when it is received or in order, adding the task to the end of the existing task queue. The LifeXplorer architecture consists of four programming and software abstraction layers as illustrated in **Figure 26**. The lowest layer can be called hardware driver layer. This layer contains the direct interface and communication channel to all hardware components. In the case of the Olympus IX81 i.e. this layer is implemented in a C++ DLL which is dynamically plugged into the MMS. On top of this layer, the "system driver connector" layer is implemented. This layer encapsulates the DLLs into a unique interface for C#. System components are provided in unique object classes such that the workflow developer does not need to know anything about the actual hardware communication protocols anymore. In this layer tasks can be pushed to the hardware layer. Whereas the hardware driver layer implements atomic tasks such as "read out image from camera", the "system driver connector" layer implements more complex tasks such as "get image stack of all available channels". The hardware control and processing layer abstracts again from the hardware tasks and encapsulates all tasks, processing and hardware control into a unique format, such that both task types are "melted" together in this abstraction layer. This layer contains the processing controller, as well as the workers, which run tasks on this abstraction level. Finally in the processing workflow logic layer, workflows can be defined visually or added in code, abstracting from how and where they are managed and executed. All the communication between distributed components, such as workers and the MMS, is implemented using WCF (Windows Communication Foundation) in order to be able to abstract from protocols (SOAP, TCP/IP, HTTP, etc.) and to

implement a SOA (Service Oriented Architecture), which was explained above. Further details on the implementation of the main components of LifeXplorer can be found in the following paragraphs.

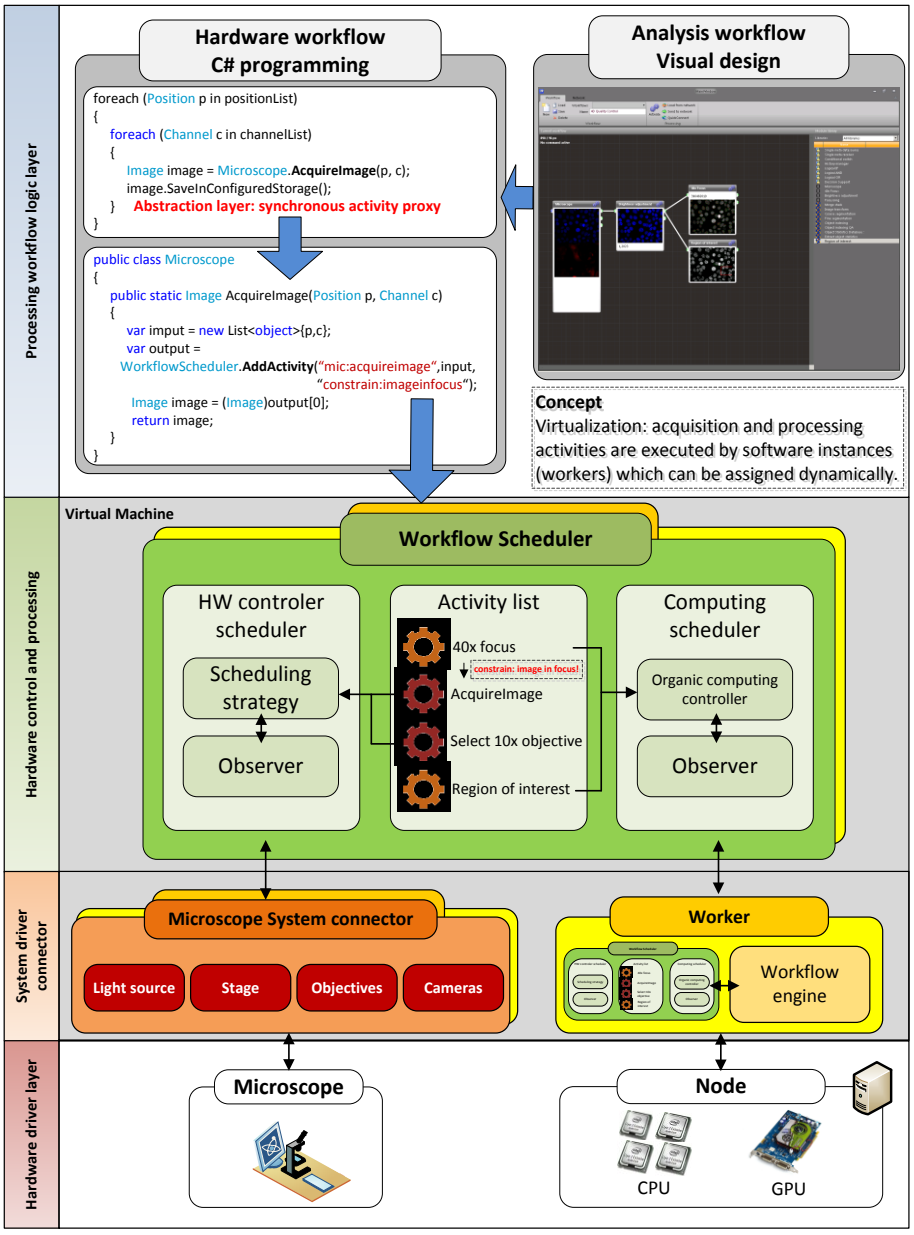


Figure 26 LifeXplorer abstraction layers. In order to hide complexity from the user and make use of parallelism and other features of high performance computing, processing and hardware tasks are abstracted in four layers such that the execution of tasks can be dynamically managed by the LifeXplorer platform during run time by the means of organic computing.

3.3.1 Modules

Modules are the top level abstraction of tasks in the LifeXplorer platform. They encapsulate the actual code. The code will normally be located in code libraries and can be used independently from the module classes and objects. Modules however consist of a standard interface, which allows defining input and output object lists. As illustrated in **Figure 27**, the module serves as a wrapper around the actual code base and can have an individual amount of I/Os.

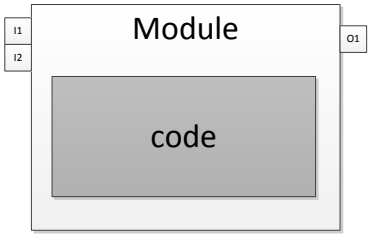


Figure 27 LifeXplorer basic module architecture

In **Figure** 28 some sample modules of applications evaluated in this thesis are presented. Module definitions need to be provided by the developer, which can easily use a given template pattern to create and define new modules. Only the IOs have to be defined and one single method execution of the program logic. Using this simple approach, external programs such as ImageJ, CellProfiler, Matlab and others can easily be linked into LifeXplorer. For the applications developed and evaluated in this thesis, different standard modules for external program integration where developed. ImageJ i.e. is called by a batch progress class, which is called inside an ImageJ interface module. The input is a matrix of short values, which represents the input image, and the output matrix contains the filtered output image from ImageJ. ImageJ can thus be easily connected by recording a macro file and providing a batch script path to LifeXplorer that is used to execute the recorded macro in a batch mode (without GUI). This also applies to CellProfiler, Matlab and Mathematica. For all these programs, standard modules are integrated into LifeXplorer. Analysis workflows can be enriched with existing standard tools and existing code without any further programming knowledge. Using .NET as the application host and runtime environment of LifeXplorer, an extended amount of integration possibilities

exists, since .NET is being permanently developed and novel sub frameworks (such as WCF, WPF, etc.) and useful technologies are provided for free.

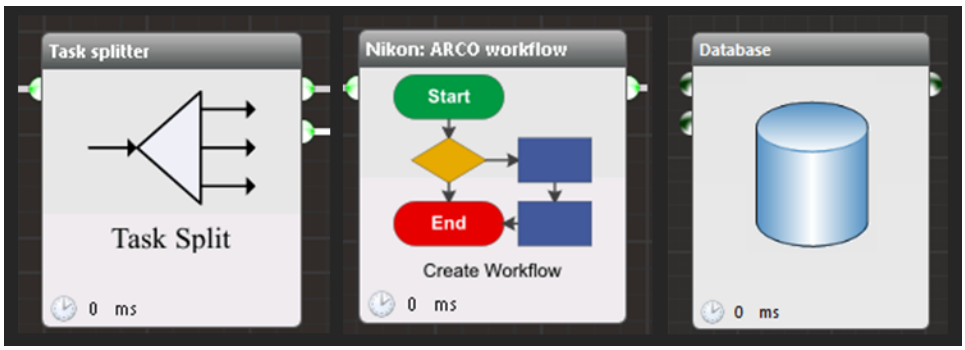


Figure 28 LifeXplorer module samples

Next to the encapsulation of complexity, LifeXplorer modules provide standard visualization interfaces. Intermediate results can be added to the primary results and will be visualized in the LifeXplorer Intelligence Manager. As the LifeXplorer code is freely available for everybody, code integration and visualization possibilities can be extended and customized. As modules are the key abstraction layer for processing and hardware acquisition tasks, they are managed by the organic computing controller, which is embedded into the LifeXplorer platform. Their execution is also permanently measured by the embedded observer components in order to ensure that the execution of a module still meets configured requirements, such as availability and latency.

Although modules are managed and dynamically associated to processing nodes during runtime, which increases the dependability of the processing and thus feedback control circuits, modules are the lowest logical abstraction, as they still implement concrete program code. As it is explained in chapter 2, ARCO serves as a basic algorithm to abstract even from concrete modules and thus specific tasks. ARCO module therefore will make use of a program class called ARCO Manager, which is able to select a specific module during runtime based on previous knowledge about the experimental system provided by the operator. These ARCO modules are a key enhancement of LifeXplorer and represent a crucial step towards higher degree of automation based on models about life that are provided to the computer.

3.3.2 Microscope Management System (MMS)

Running an automated microscope experiment demands a graphical interface for operators, because they are usually non-experts in microscopy and programming. Further, it is more dependable and efficient to set up experiments graphically. As illustrated in **Figure 29**, the author therefore developed a microscope management system (MMS), where operators can set up microscopy workflows for multichannel and multipoint experiments. Obviously there is a large amount of existing microscopy software available already. The development of the MMS, however, was necessary because its purpose was to be connected to an accelerated high-throughput microscope, a modified Olympus IX81. The acceleration of this system was done by fixing and optimizing the optical light paths, using a multi-camera setup, and by hiding latencies of the camera readout and stage movements. Both optimizations influenced the design of the hardware driver and the workflow, regarding how to capture images and perform multipoint experiments. In order to keep this chapter as short as possible, further details of this acceleration approach will not be explained. However, the high-level acquisition workflow is unfortunately directly influenced by advanced techniques to accelerate the acquisition process. In addition, accelerated focusing methods were developed and evaluated throughout this thesis. This furthermore required a microscopy framework in which all software layers could freely be implemented and optimized, which was the reason to develop the MMS. Eventually the service-oriented architecture approach of LifeXplorer, applying intelligent microscopy methods, was a major reason to use .NET and build a new microscopy framework. The MMS both serves as a development framework for novel microscopy methods as well as a graphical user interface for standardized routes of imaging. Being connected to the organic computing network via a standard interface of the workers' service oriented design, the MMS smoothly integrates into the LifeXplorer's architecture. Integrating new feedback signals i.e. such as region of interest (ROI) repositioning, or a refocusing trigger, the MMS's scheduler behavior for resynchronizing new hardware tasks can also be set up in the user interface. As the MMS itself is connected by a worker to the

computing network, (and especially to a worker which runs the virtual microscope module) runtime information about the feedback control messaging and real-time data analysis can be debugged and run time information be added to a central logger, as it is illustrated in **Figure 29**. The processing monitor directly visualizes run time information that is added to the logger. The logger can receive information from every processing stage, from the low level driver stages to the high level organic computing messages and decisions.

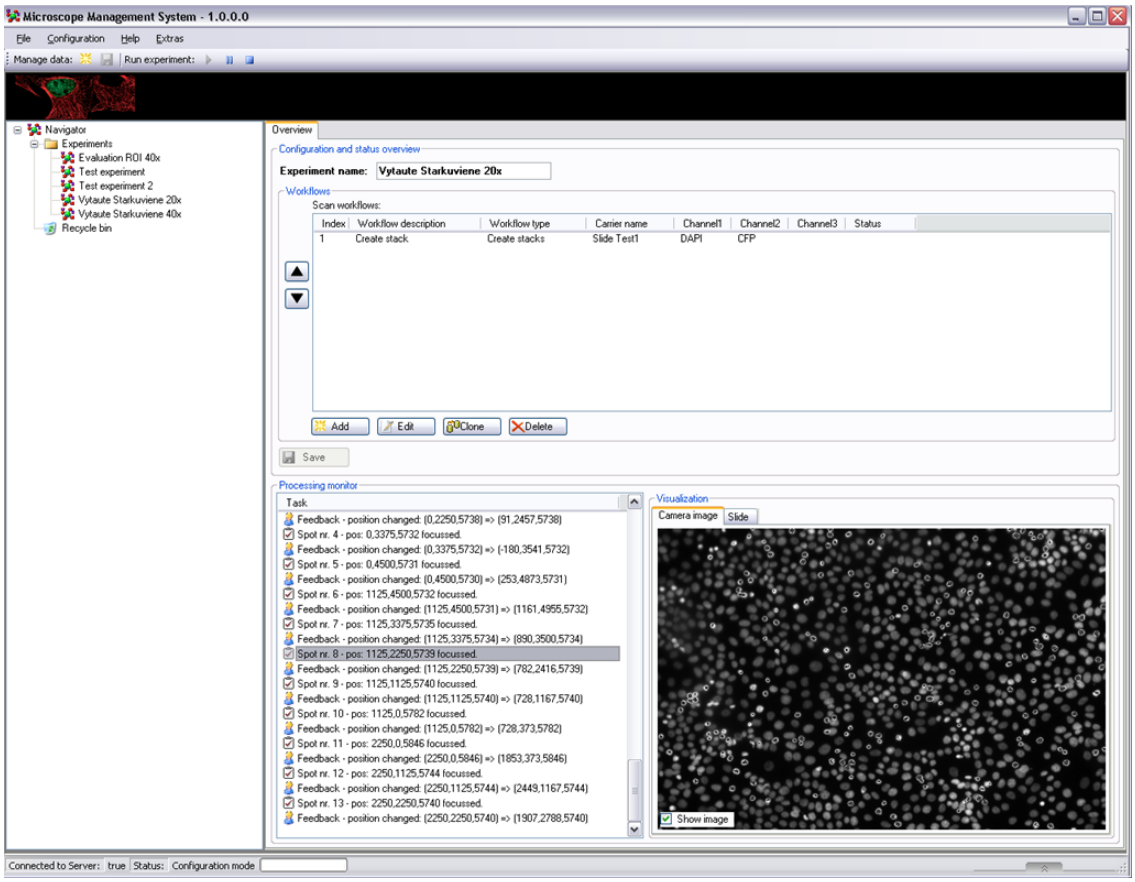


Figure 29 LifeXplorer's Microscope Management System

Because the system was designed also to optimize data processing for heterogeneous computing networks, workers present their available resources and functionalities to the MMS once they connect. As illustrated in **Figure 30**, once a worker connects to the MMS, it will transfer its available resources and functions to the MMS. The MMS then can decide which workers to use for which computing task (ROI, etc.). The virtual microscope module, which was explained in the architecture, is normally configured to be run only on the MMS host PC, from where a n:n data distribution is started. The MMS however is built to

handle workers directly without even knowing about the organic computing manager and other components of the LifeXplorer's computing platform. It is able to implement redundancy and thus high availability. This direct feedback loop implementation however describes its basic functionality, and was later extended by means of organic computing, implemented in separate components such as an observer and a controller unit. By means of organic computing, processing tasks can be allocated dynamically on run time to different nodes. The basic approach however is that the microscope (or rather the MMS) will check if a worker is able to run a processing task or not, to ensure that the initial data distribution goes to the right processing node. In the case of the LifeXplorer's organic computing network, the virtual microscope module holds an instance of all functions available in the network, and in doing so can virtually offer processing tasks to the MMS. The actual task execution however can be performed on any computing node.

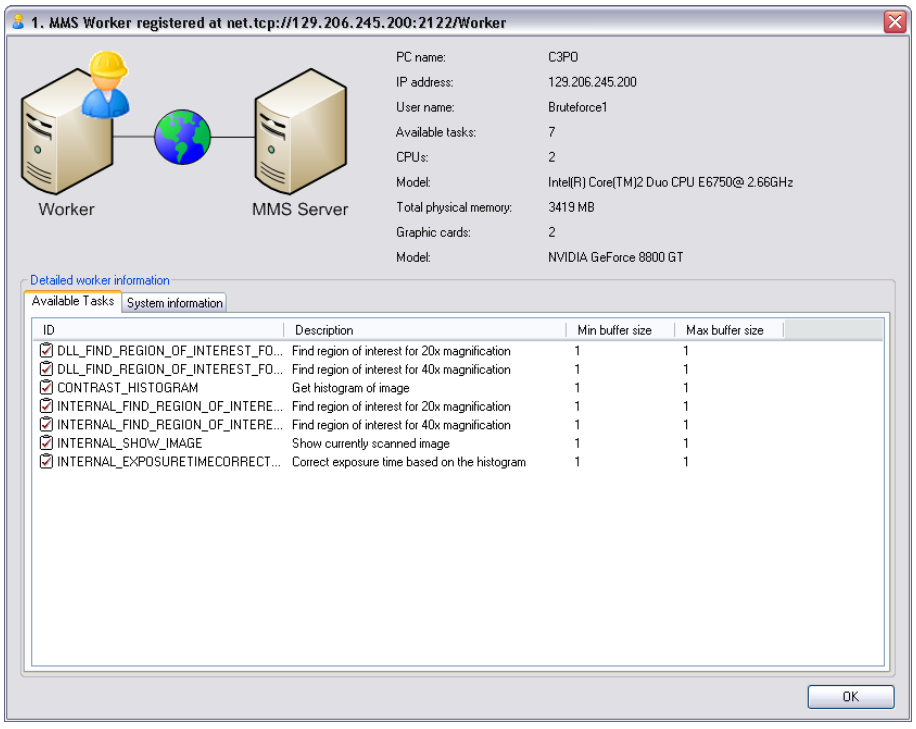


Figure 30 MMS worker resource information dialog

Again the MMS in the case of the dialog presented in **Figure 30** serves as a development environment and offers run time and information processing.

3.3.3 Workers

An important design approach in the LifeXplorer was to abstract processing nodes with a software component called “worker”. A worker builds a multipurpose processing instance and in the latest version of the architecture receives tasks and the corresponding program code on demand. **Figure 31** illustrates the basic architecture inside a worker. Having the service oriented architecture (SOA) paradigm in mind, a service is a software component based on object orientation that offers its functions to other services over the network. By that approach a SOA is by default scalable, meaning that functions embedded in its services can easily be spread from a single computer to a large network of computers. The software itself will stay exactly the same, only the bindings between a service client instance and a network address needs to be updated. The architecture of a worker therefore essentially needs to have a network interface, which is able to communicate processing results. Furthermore, related to the SOA paradigm, this interface has to be able to even communicate structured data (like objects etc) by default, without the need for explicit protocol programming. Objects on run time are dynamically resolved therefore using i.e. XML description of the object structure. To communicate whole object structures, a serializer on the sender and deserializer on the receiver side are necessary. Several serializer and deserializer classes are implemented by default inside .NET and can be used in the Windows Communication Foundation (WCF), a powerful communication framework.

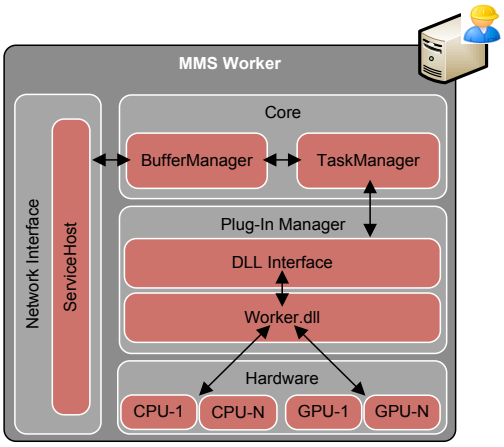


Figure 31 Worker architecture

.NET and WCF were used to standardize communication and even be compatible with other platforms through the SOAP (Simple Object Access Protocol), which is an industry standard for SOAs. SOAP however is based on XML, which makes the SOAP protocol by default slower, because for one the serialization and deserialization process is more complex, and the protocol overhead is also higher than for a usual binary protocol. For the internal communication in LifeXplorer, however, a rather classical approach was implemented to boost the serialization and deserialization process for communication. A binary memory stream / block serialization was used which does not allow for dynamically assigned object structures on the receiver side, but only allows to deserialize the whole object structure in one step, and cast the object pointer explicitly to the actual object type behind the memory stream. This boosts latency and communication throughput, and still allows the architecture to behave as a SOA on the level of workers and available processing tasks.

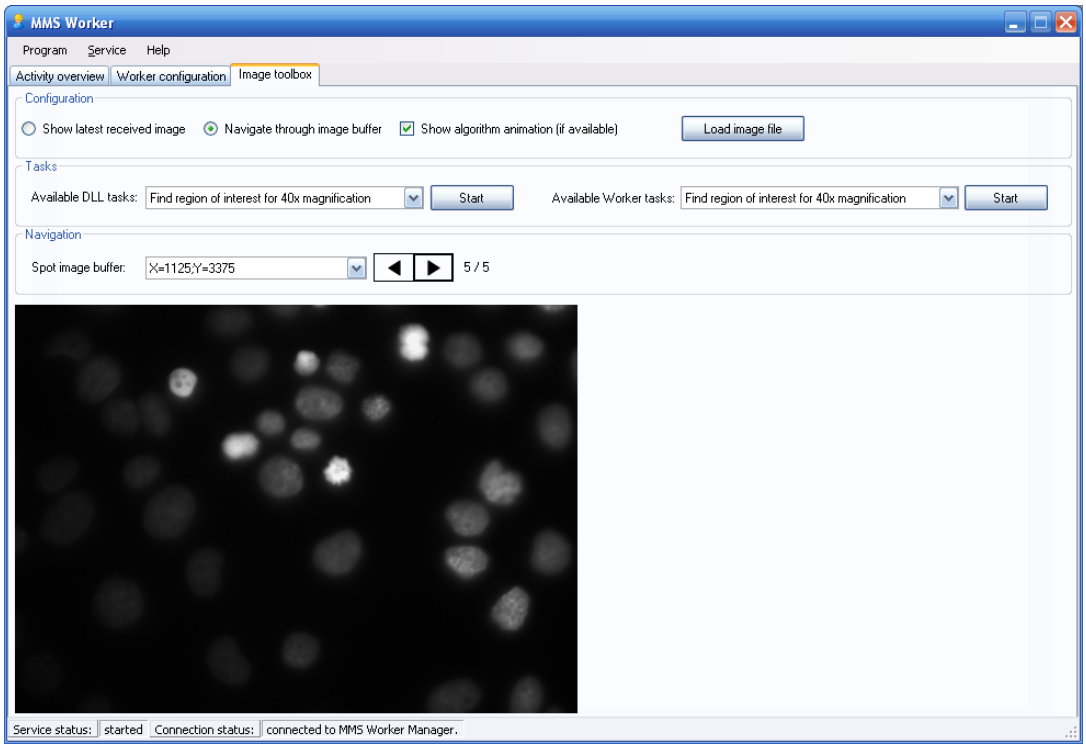


Figure 32 Worker graphical user interface with run time information

In order to more efficiently achieve a better understanding of run time behavior of a feedback control system implemented in LifeXplorer, the workers offer basic data visualization as is illustrated in **Figure 32** and **Figure 33**. Exemplary in **Figure 33** a region of interest algorithm is run and the output XY-vector to optimize the imaging position is presented directly in the worker to provide more transparency to the feedback control developer on run time.

Within the worker a task management is implemented as is presented in **Figure 31**. Directly after the service host receives a task, a buffer manager is implemented which buffers and organizes the input data. Inside the worker's core the task manager finally executes tasks, which are provided by the buffer manager. The task manger manages the task execution. The task manager however not directly executes the program code behind a task, but calls a plugged in method, which is located in an external library, i.e. DLL. This library code then finally executes the program code for a certain processing task. Depending on the implementation, the program code can be run on different locally available resources such as CPUs or GPUs.

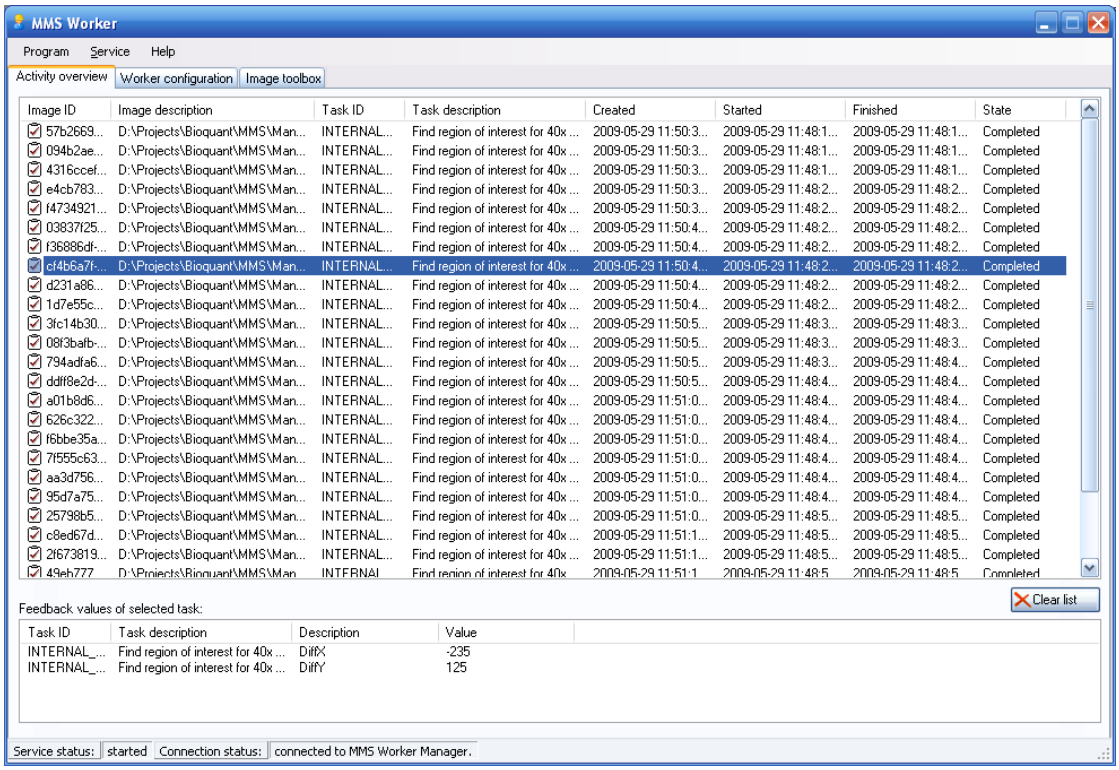


Figure 33 Worker task execution log

3.3.4 Intelligence Manager

Feedback controlled microscopy couples image acquisition and data analysis in a feedback control loop. Based on real-time data analysis, decisions about the screening workflow are made during run time. Developing such control circuits and decision support systems, however, is complex. Especially when the data analysis itself is complex already, it becomes more complex when real time data needs to be processed. Visualization of intermediate processing results, together with run time information, is a key to efficiently develop, test, and optimize such applications of intelligent microscopy. Especially when dealing with image processing results, where the optical impression of results serves as a fast functioning quality assurance test. Since LifeXplorer was developed as a general-purpose platform for a large variety of science automation applications – i.e. applications of intelligent microscopy - a graphical user interface as illustrated in **Figure 34** was an added value towards user-friendly configuration of LifeXplorer's "intelligence".

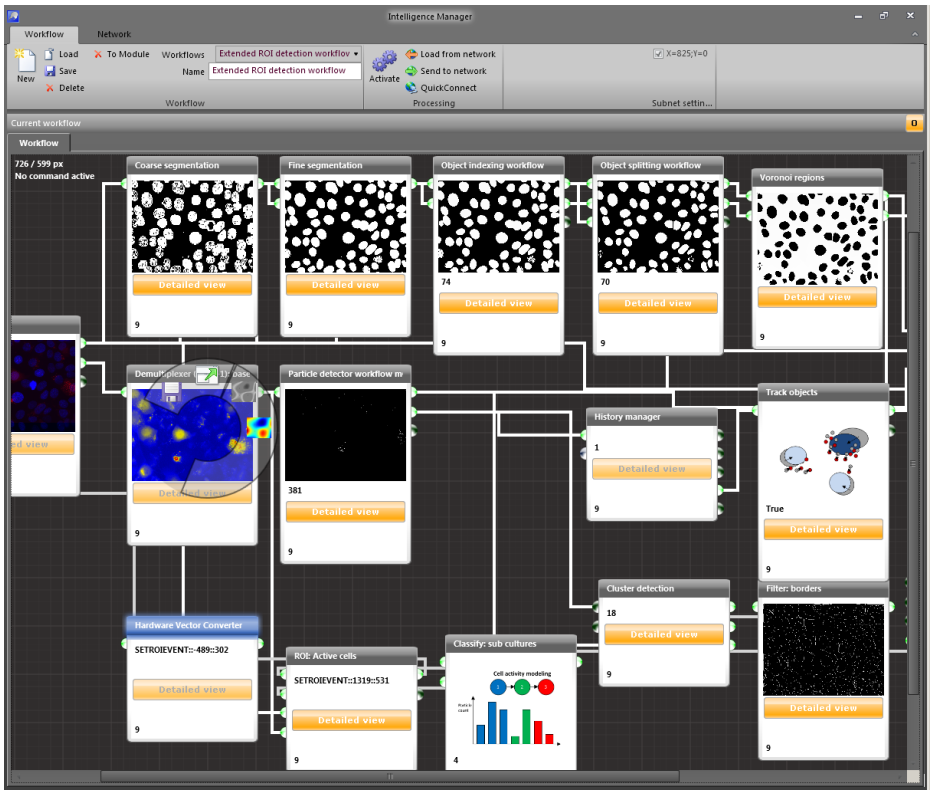


Figure 34 LifeXplorer Intelligence Manager

Complex data analysis workflows and circuits for feedback control can be visually programmed as illustrated in **Figure 35**, showing the complete circuit for the intelligent light exposure control methods presented in this thesis.

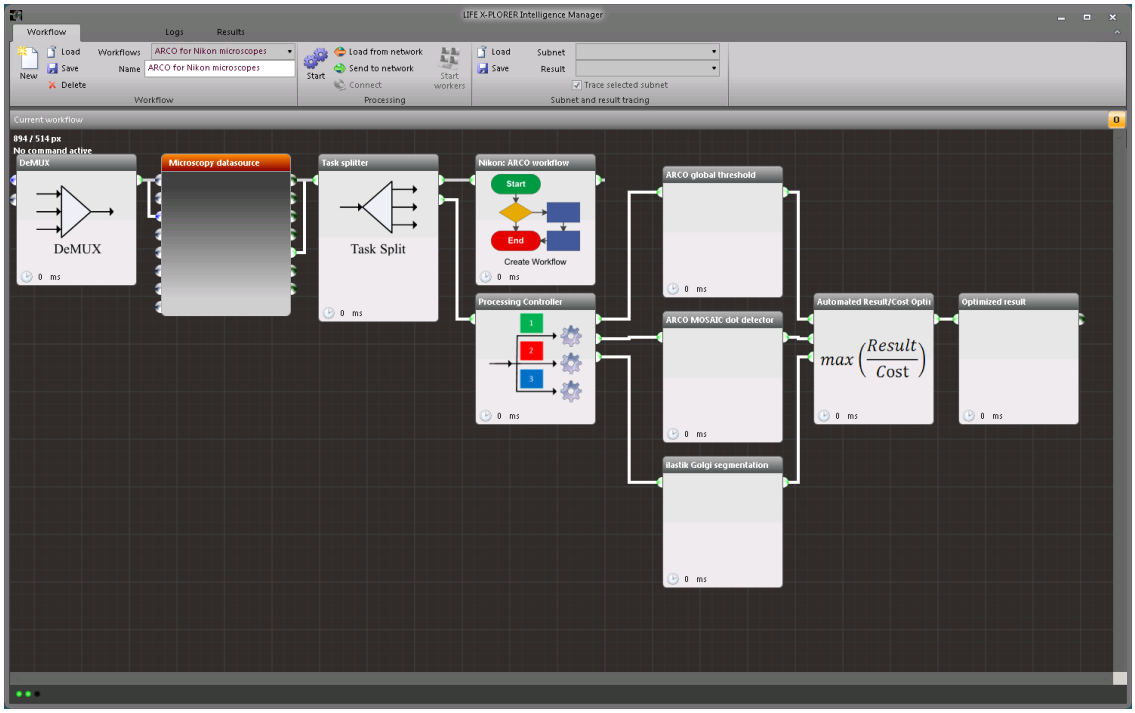


Figure 35 LifeXplorer ARCO circuit for intelligent light exposure control

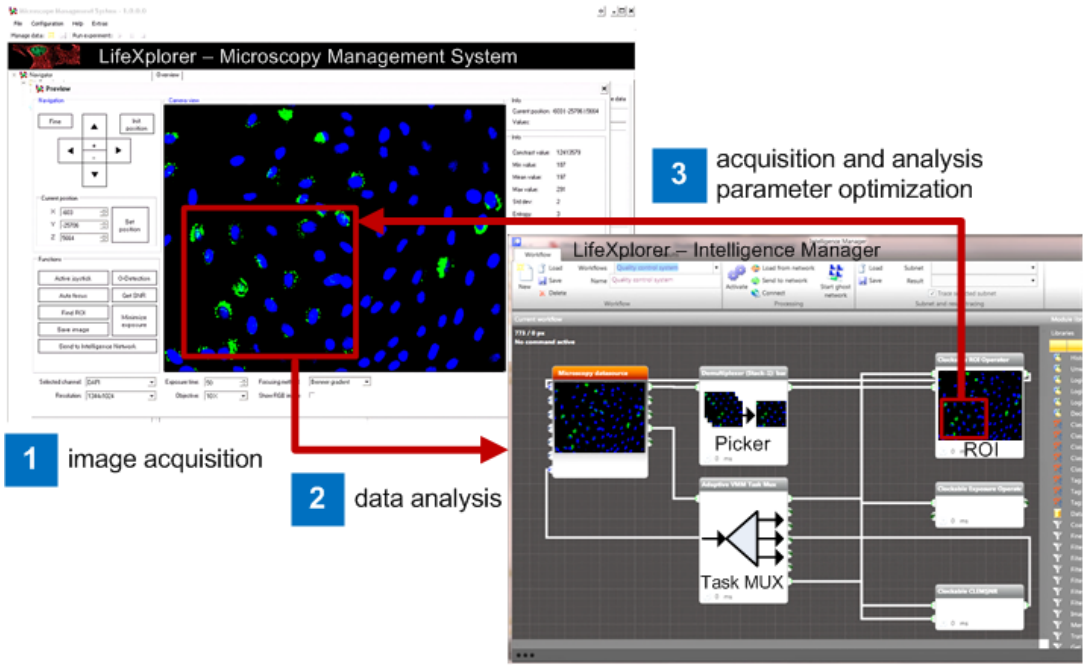


Figure 36 LifeXplorer's acquisition and processing framework in a feedback loop

3.3.5 Organic Computing Service

To distribute processing tasks, the LifeXplorer platform implements mechanisms of self-organization (self-x) [42]. Due to better complexity management and maintenance of existing components, services are used as a base technology for LifeXplorer. The organic computing service (**Figure 37**) inside LifeXplorer implements the logic and strategies to distribute processing tasks to the computing network. As mentioned above, processing nodes are virtualized using a multipurpose software component called "worker". In contrast, however, to classical SOA implementations, LifeXplorer arranges workers and their functions on demand based on a peer-to-peer connection. That means that workers can get assigned new functionality, by sending the corresponding program libraries to them. The organic computing service serves as a component that is able to use workers, which can be run locally or distributed on a network, to build a communication topology such that a given processing workflow can be run following a configurable process optimization strategy. Samples of different process optimization strategies are presented in **Figure 39**.

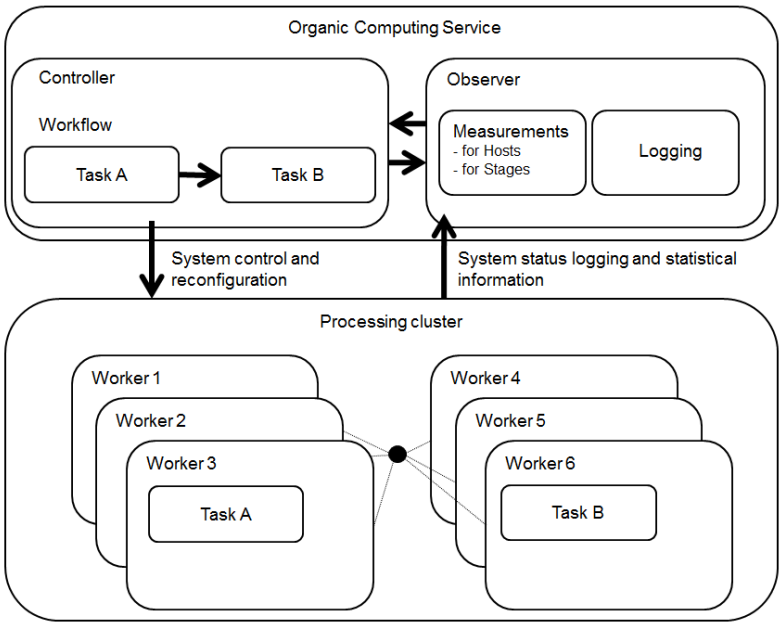


Figure 37 Organic Computing Manager architecture

The organic computing service' components are illustrated in **Figure 37**. In the intelligence manager interface, operators can set up a data analysis or feedback control workflow. This workflow has to be sent to the "network" as the button in the interface says. By that it will be sent to the organic computing service and forwarded to the embedded controller component. The controller, based on different strategies (**Figure 39**), will then distribute single tasks or task packages of the workflow to different workers as illustrated in **Figure 38**. Different processes can be run on one worker using the same function and libraries. The organic computing service sets up the processing topology by sending a list of tasks objects to a worker, and by sending an output routing list. The routing list consists of task object ids and receiver addresses. Once the controller has updated this routing information, workers are connected directly with each other and will send results in peer to peer mode. If the observer measures that a worker is not available anymore, or the processing latencies are too low, the controller can update the routing information and in doing so dynamically optimize the processing network topology during run time.

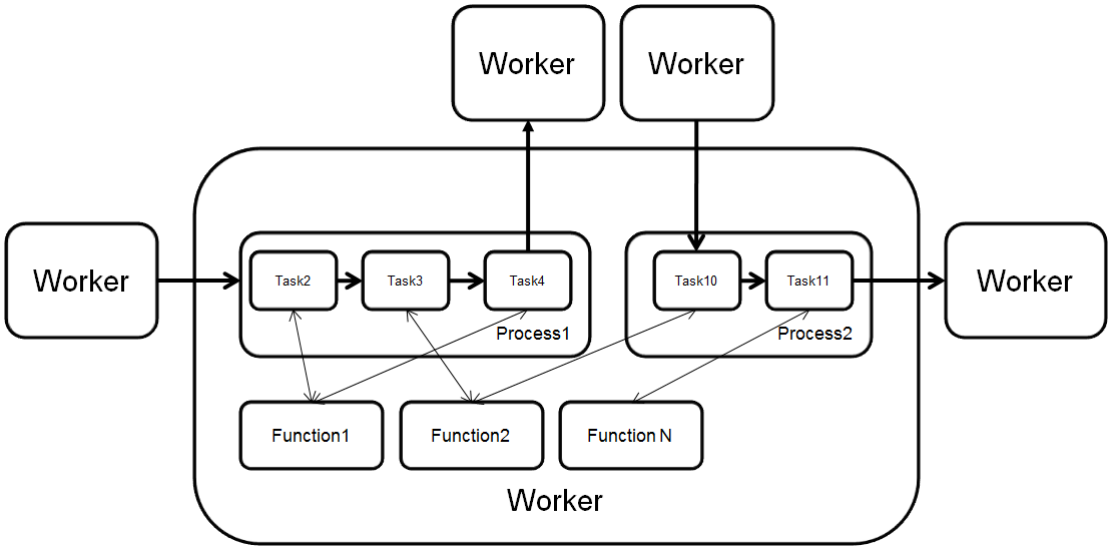


Figure 38 Organic computing: process distribution sample

For the task distribution not only the configured scheduling strategy as illustrated in **Figure 39** is relevant. Important are also the preconfigured binding strategies, as will be explained in the following chapter. The most easy and inflexible binding strategy is the fixation of a task to a single processing

node. All instances of this task within a workflow would then need to follow that rule and use the one node for the task is was set up for, independent from the distribution strategy. Different scheduling strategies are implemented in LifeXplorer. As illustrated in **Figure 39** a min, max resource usage strategy is implemented as well a "min time to return". By default the max resources strategy is used in LifeXplorer. All strategies implemented in LifeXplorer are based on simple parallelism of the whole workflow, meaning that new processing packages coming from the virtual microscope unit are sent and processed as a whole on a single node. How many and which nodes are used depends the strategy and observation of their performance. The min resources strategy is based on a dynamic expansion of the processing topology, once a certain local latency threshold is overrun.

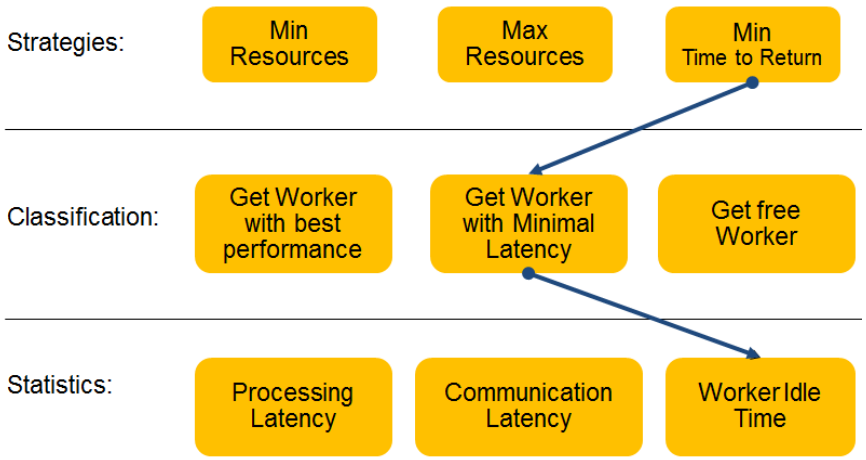


Figure 39 Organic computing scheduling strategies

3.3.6 Database

The database of LifeXplorer is based on the database management system Microsoft SQL Server (2012). It was designed for large data analysis of single cell population wide analysis and time resolved data for multipoint and multichannel experiments. LifeXplorer offers feedback control and dynamic decisions for every analysis step. After the image processing is performed on multiple sampling positions, single objects and object features are extracted and have to be put into relation in a global statistic. As LifeXplorer is supposed to offer intelligent experiment control and analysis for a wide range of applications, it offers to the developer a unique interface for feature extraction and statistics. This interface is mainly based on a general ImageObject class. ImageObjects can be added to the database via a corresponding database module. All statistics in this thesis were created using this interface and the database queries to create complex statistics. Nuclei as well as Golgi complexes and mitochondria dynamics were abstracted into objects and put into relation. Single cell statistics i.e. require the relation between primary statistical objects such as Golgi complexes and single cell information such as the nuclei. Using i.e. Voronoi regions, particles can be related to nucleus objects. Tracking a large amount of single objects over time required a scalable processing approach and a suitable access for complex data analysis queries. Both are provided by the SQL Server and the embedded stored procedures that were developed to perform single cell and population wide dynamics analysis. **Figure 40** illustrates the first part of LifeXplorer's database design. The main table is the OBJECT table. Two additional tables for classification, OBJECTTYPE and OBJECTCLASS are related to this main table, as well as a CHANNEL table. For object tracking i.e. the table PARAMETER_TRACKING_FUZZINESS is related to the OBJECTTYPE TABLE. This allows i.e. defining different fuzziness configurations for different objects, i.e. nuclei and particles. With the table OBJECTCLASS classification information can be related to each object. I.e. the phenotype can be an object class, or provide finer information on parameter classifications, such as size, brightness, etc. The table RELATION_OBJECTCLASS allows an n:n relation,

which means each object can be related to several classes. The LifeXplorer database module automatically creates object classes, before adding new objects to the database. Object classes are thus synchronized to the database at a central position, whereas each classification module can freely add new classes to the objects directly, without the need to know about the statistics layer implemented by the database. A special processing scheduler built into the database is directly related to the main object table OBJECTCLASS.

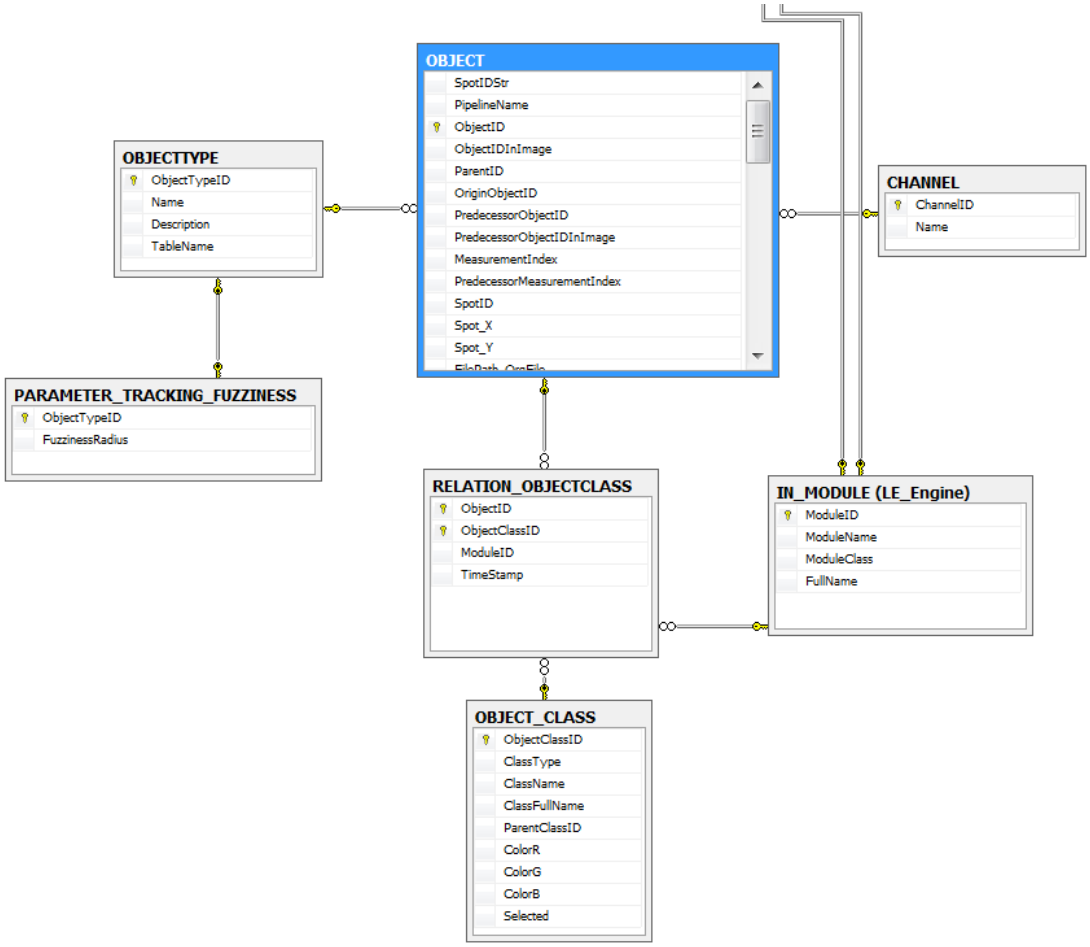


Figure 40 LifeXplorer's database design for time-resolved object statistics

The table IN_MODULE is related to the table RELATION_OBJECTCLASS, because of two reasons. The first reason is that each module can potentially add new classes to an object. The second reason is that classes themselves can be clustered into the modules they were created by. This helps to schedule processing tasks for efficiency. **Figure 41** illustrates the

database design part for the parallel processing scheduler, embedded into the database. The reason for implementing a separate processing scheduler at this analysis level is that in some cases it can be faster or easier for the developer to run experiment wide statistics on the program source code level. As each sampling points can be processed on a separate processing node, processing nodes have to have a central place where they can add results of different sampling positions. One experiment i.e. can be run on several sampling points, in order to have enough cells for a robust statistic. This then implies that the objects are added to the database and re-synchronized to a processing node, which creates experiment-wide statistics. LifeXplorer offers both options: either to run statistics directly in the SQL Server using SQL and stored procedure code, or to use a module that is classifically run on a worker instance and create statistics using normal programming code.

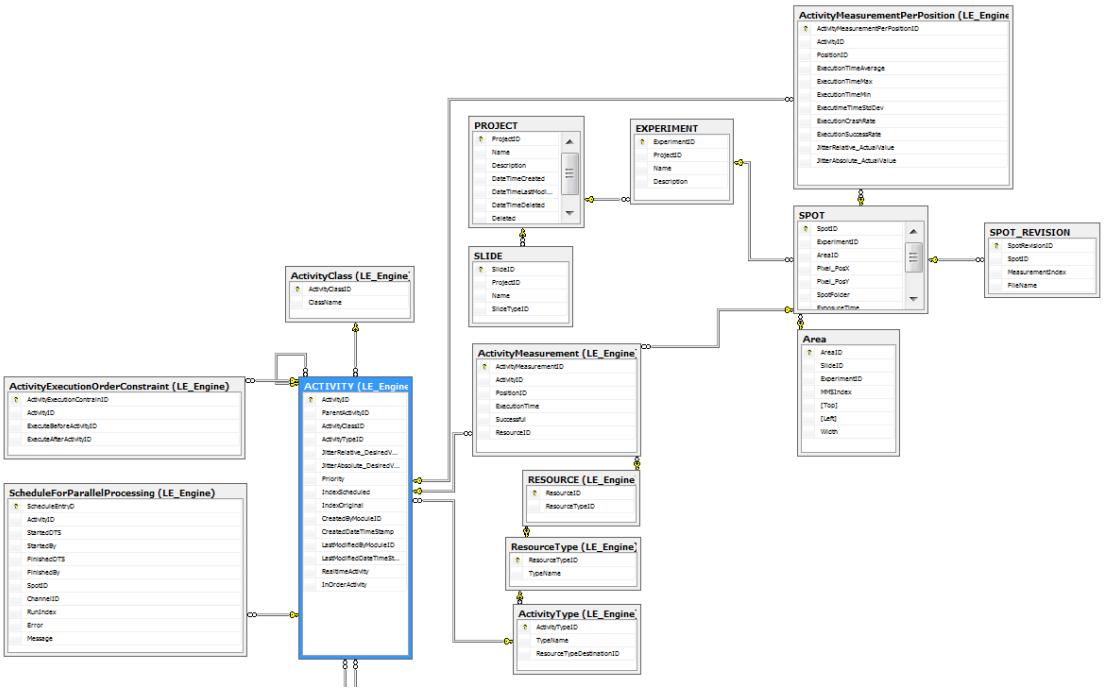


Figure 41 LifeXplorer's database design for parallel processing and activity scheduling

To provide the second option however, the database design illustrated in **Figure 41** was designed to provide basic scheduling functions for parallel processing. Objects that should be processed can be packed into several processing packages. For example, a node can request all objects of a certain channel of a certain imaging run number. The scheduling then will block this processing package for a

single node so that the next node in the network requesting a processing package will get a different processing package. Objects that are processed in this way can be blocked, so that statistics can be created exclusively, and in order. Several processing package levels are available. **Figure 41** illustrates that the database design is built around the table ACTIVITY. And activity is a task that is run with a certain configuration and data. The task could be nuclei tracking, whereas the configuration is that all nuclei of a certain measuring index should be tracked. This activity is then added to the table ScheduleForParallelProcessing. Activities can be executed in order, meaning that a certain activity relies on the scheduling that another activity is running before its execution. The table ActivityExecutionOrderConstraint holds the corresponding information. All activities in the LifeXplorer network are measured. In the table ActivityMeasurement, performance measurements of activities can be saved. These measurements as mentioned above are used for the organic computing scheduling; i.e. deciding which algorithm is run on which processing resource. Eventually **Figure 41** illustrates the basic design for an experiment. A project can be set up with different experiments, slides, spots and areas. This information is provided by the Microscope Management System experiment definition. Every table in this definition also builds an information aggregation level. I.e. the experiment can, as mentioned above, aggregate different spots. Each spot can have a large number of single objects over time, which are located in different optical channels. Objects can have types and classes. The scheduling as mentioned above can consider these aggregation levels by creating object packages.

3.3.7 Plugins for Image Processing and Hardware Control

Building the applications evaluated in this thesis, the author created several modules divided into eight plugins to extend LifeXplorer's functionality. Most of the modules were used for the ARCO applications described in this work. The modules are briefly explained in the following.

Base library plugin

- AddSamplingPoint
  Adds a sampling point to the Microscope Management System scheduler.
- AND
  Logical, binary "and" operation with a configurable number of inputs.
- Decision support
  Freely programmable decision support logic with dynamic inputs. Programming code can be added and modified during runtime.
- DeleteSamplingPoint
  Deletes a sampling point from the Microscope Management System scheduler.
- History manager
  Caches any kind of input over time with a configurable depth. Outputs the last value (object) as well as a stack of all saved values.
- IF
  Logical, binary "if" operation with a configurable number of inputs.
- OR
  Logical, binary "or" operation with a configurable number of inputs.
- SetExposureTime
  Sets exposure time for all sampling point or a selected one.
- SetSamplingRate
  Sets global sampling rate.

- SetXYPosition
- Modifies XY position for a given sampling point
- SyncSamplingPointPositions

  Modifies the list of sampling points for in the Microscope Management System.
- SyncSurfaceModel

  Synchronizes and if configured creates a calculated surface model with the Microscope Management System.

Classifier plugins

- Classify: crumpled cells

  Finds and tags crumpled cells, based on outline to size ratio.
- Classify: object brightness

  Classifies object brightness based on histogram into five classes: very dim, dim, average, bright, very bright.
- Classify: phenotypes

  Classifies objects into phenotypes based on the trainable ilastik classifier.
- Classify: round objects

  Finds and tags round cells, based on outline to estimated radius ratio.
- Tag: objects at border

  Tags objects the border of an image.
- Tag: objects invalidated by channel merge

  Tags objects that need to be invalided because of a channel merge that creates a non overlapping area.

Database plugin

- Database

  Database module that adds objects of the class ImageObject to the database (MS SQL Server). Three inputs (objects of three channels) can be configured to be synchronized dependently or independently.

Filters plugin

- 4-tiles video creator

  Creates video frames. The video frames consist of four tiles, each showing RBG images of one channel over time. The first tile shows the latest frame, the second the difference image to the last tile (depending on a configurable sampling rate). The third tile shows the very first video frame and the forth the difference of the current frame to the very first one.
- Background subtraction

  Subtracts the background of an image. The background is calculated with a median filter of a configurable kernel size.
- Coarse segmentation

  Gradient based segmentation algorithm with global thresholding.
- Filter: by object parameters

  Sorts out objects by a configurable threshold of a configurable object parameter.
- Filter: binary filter

  Binary filter based on a global threshold.
- Filter: borders

  Sorts out all objects at the border of an image.
- Filter: cross correlation

  Creates the cross correlation of two images.
- Filter: Laplace convolution

  Filters the image based on a Laplace convolution.
- Filter: Mean convolution

  Filters the image based on a mean convolution.
- Filter: Median convolution

  Filters the image based on a median convolution.
- Filter: smart threshold filter

  Filters the image based on a smart threshold filter. Smart here means that the global threshold is automatically detected by a histogram analysis.

- Filter: stamp out binary map

  Sets all values to zero, where a reference image has values greater than zero.
- Fine segmentation

  Based on a coarse segmentation as mentioned above, local thresholds are determined and used to segment i.e. nuclei in more detail.
- Fire LUT

  Creates an RBG image based on a so called fire look up table: small values are mapped to blue, high values to red.
- Get image statistics

  Creates image statistics (average value, stddev, histogram, etc.)
- Image transform

  Transforms an image based on a configurable transformation vector.
- Maximum projection

  Performs a maximum projection for a stack of images.
- Merge stack

  Merges an image stack of size n into one image by calculating the average value of each pixel.
- Remove background by subtraction (on stack)

  Uses the image with the highest contrast and subtracts a blurred image as the background.
- Stitching unit

  Stitches a grid of images into one single image, i.e. a 10x10 grid into one image.
- Transform object positions

  Transforms object positions based on a transformation vector.

Microscope Control Logic

- Automated Result/Cost Optimization

  Sample module for the ARCO algorithm applied on light exposure and object segmentation.

- DeMux

  Has multiple trigger signal inputs and can be connected to the unique trigger input of the virtual microscope module.
- Highest quality picker (3-1)

  Choses the image with the highest quality (different measurements possible) out of a stack with three planes.
- Highest quality picker (Stack based)

  Choses the image with the highest quality (different measurements possible) out of a stack with a dynamic number of planes.
- Identify segmentation efficiency

  Counts pixels of a segmentation map which overlap with a reference map.
- Matrix buffer

  Caches a matrix and gives back the cached matrix at the output.
- Microscope

  Virtualizes in- and output of the microscope. Serves as the microscope interface for LifeXplorer's computing network. Handles redundant, parallel trigger signals and implements other special functionalities.
- Nikon: ARCO workflow

  Creates a macro files for Nikon microscope systems that images the same position with different light exposures, based on an exposure time that is read out during runtime and a configurable sampling step.
- Nikon: XY-readjustment

  Re-adjusts the XY-position of sampling point by maximizing the overlap of a segmented object map with a stitched 10x10 reference image.
- Optimized result

  Provides the image with the maximal result / cost ratio.
- Refocus Trigger

  Provides a trigger signal to refocus a positions once the quality of an image at this position dropped under a configurable threshold.
- Save to disk

  Saves image to disk at a configurable file path.
- Task splitter

  Sends clock signals at different outputs depending on the input task type.

Scheduler

- Post processing scheduler
  Provides object packages based on a configurable package configuration (measurement index, channel, object type or class, etc.) for parallel processing of object statistics.

Object detection

- ARCO based segmentation
  Sample module for the ARCO algorithm applied on object segmentation for all algorithms added to the object detection manager.
- ARCO global threshold
  Automatic optimization of a global thresholding segmentation pipeline by the means of ARCO.
- ARCO MOSAIC particle detector
  Automatic optimization of the MOSAIC particle detector by the means of ARCO.
- ARCO ilastik object classifier
  Automatic optimization of the ilastik classifier added to CellProfiler by the means of ARCO.
- AssignParticlesToCells
  Assignes particles to cells based on a particle, nuclei and Voronoi map.
- AutoConfigParticleDetector
  Outputs optimized parameters for the MOSAIC particle detector by the means of ARCO.
- AutoThreshold
  Provides an optimized global threshold based on a histogram analysis.
- CellProfiler_Ilstaik_GolgiSegmentation
  Golgi segmentation using ilastik and CellProfiler.
- CellProfiler_Ilastik_NucleiSegmentation
  Nuclei segmentation using ilastik and CellProfiler

- CellProfiler_ParticleDetector
  Particle detector using CellProfiler.
- CenterOfMassOfSpecificPhenotype
  Determines the center of mass for configurable objects types.
- ClusterDetection
  Extracts object clusters by a configurable average neighbor distance.
- EstimateRadiusForBorderCells
  Estimates the radius for nuclei at the border of an image.
- ExtractObjectStatistics
  Extracts basic object statistics and morphology information.
- GlobalThresholdSegmentation
  Segmentation filter based on global thresholding.
- ImageJ_LoG3D
  Laplacian of Gaussian filter, implemented in ImageJ.
- ImageJ_Mosaic_ParticleDetector
  Mosaic particle detector, implemented in ImageJ.
- ImageJ_SegmentationPipeline
  ImageJ segmentation macro (can be customized).
- ImageJ_Watershed
  ImageJ watershed algorithm.
- Matlab_MexHatWavelet
  Mexican hat wavelet transformation implemented in Matlab.
- Merge3DObjectData
  Merges object data that is located in different z-planes, by choosing overlapping objects with the highest variance (intensity dynamic).
- ObjectQA
  Quality assurance for objects, based on min and max size and a min standard deviation.
- ObjectIndexing
  Indexes objects in a binary map.
- ObjectIndexingQAErase
  Combines ObjectIndexing and ObjectQA

- ObjectIndexingWorkflowNuclei
  Complete object indexing workflow for nuclei objects.
- ObjectSplitting
  Splits objects by eroding them, until new objects are found or a maximal run number is reached.
- RegionOfInterest
  Maximizes the number of objects with a specified tag (class or object type) within a region of interest (size can be configured).
- ROIBasedOnNucleiCount
  Uses RegionOfInterest to maximize the nuclei count.
- SingleCellParticleDector
  Complete workflow for single cell particle detection.
- SingleCellStatistics
  Complete workflow for single cell statistics.
- VoronoiRegions
  Creates a Voronoi map based on an object index map

Tracking

- Track objects in DB
  Two-dimensional tracking of objects over time based on least squares, implemented as a stored procedures in the database. Works incrementally with a configurable depth (if objects may get lost over time and reappear).
- Track objects
  Two-dimensional tracking of objects over time based on least squares. Works incrementally with a configurable depth. Can be run in parallel for multipoint experiments, as the implementation is run in LifeXplorer's computing network.

Chapter 4
Applications: of the ARCO algorithm

4.1 ASEC (Application Specific Exposure Control)

4.1.1 Abstract

Phototoxicity [5, 6] as well as photobleaching [7-9] pose serious challenges in live cell imaging, causing cells to react abnormally or to become apoptotic much earlier than they would without being measured with light. Image analysis is mostly interested in specific features like cell count, nuclei segmentation, object texture, etc. The exposure time is generally set to a best guess by the operator. I propose to automatically re-adjust the exposure times during the experiment such that the feature extraction achieves a required accuracy, and at the same time minimizes the light exposure as well as the photo bleaching and phototoxicity effects. The method called ASEC (Application Specific Exposure Time) is directly derived from the ARCO algorithm and an application of ARCO in the experimental parameter dimension light exposure. In addition the method dynamically re-adjusts over time, which also can be derived from ARCO. Two different biological systems were used to prove the added value of ASEC for live cell imaging. In both cases it could be quantitatively shown that the light exposure can be minimized to 90% of the operator's subjective estimation, that it reduces phototoxicity three- to six fold, and it allows three- to sevenfold longer observation time on living cells.

4.1.2 Introduction

Fluorescence live cell imaging has become a relevant application in life sciences and systems biology. Observing living cells however leads to complex problems. Next to the experimental set-ups themselves, the process of reading out signals from the cells with light microscopy poses a variety of challenges. One the most challenging issues are the so-called phototoxicity [5, 6] and photobleaching [7-9] effects. Phototoxicity is the phenomenon of toxic effects caused by photons interacting with molecules in the cells. Mainly excited fluorophores produce reactive oxygen species (ROS) [33, 34, 43, 44, 112, 113]. ROS can react with a variety of oxidizable components, such as nucleic acids, lipids, fluorophores and proteins, which leads to a loss in the fluorescence signal (photobleaching) and can cause cell cycle arrest or finally cell death (phototoxicity). The production of ROS however is mainly dependent on the light dosage, based on the photochemical properties of the fluorophores [14]. Minimizing the light exposure is an effective way of reducing phototoxicity and photobleaching [6, 10, 114]. Reducing the light exposure however negatively affects the quality of the resultant images. The information content that can be extracted is correlated to the dynamic of the image. The less light exposure is used, the lower the signal to noise ratio (SNR) [115, 116]. This means that it becomes more complicated to extract information, highlighting a conflicting goal between image quality (high light exposure) and cell viability (low light exposure). Previously, reduced illumination has been achieved via optimized pixel-dwell time with specialized laser scanning microscopy [10, 11, 35-37], and through development of application-specific refocusing algorithms [1-3]. Controlled Light Exposure Microscopy (CLEM) [10] for laser scanning microscopes showed that the phototoxicity and photobleaching effects in imaging can be reduced if background pixels are exposed with a different exposure time as foreground pixels. How the light dose is influencing the viability of cells was also investigated recently, although no established standard exists to approximate phototoxic effects quantitatively [6]. To solve this problem more generally and find the most accurate configuration, ARCO can be integrated into a light exposure control

system. ARCO is able to identify how much light is necessary to still gain most of the predefined information. We called the ARCO based light exposure control method ASEC (Application Specific Exposure Time). ASEC controls the light exposure to reduce phototoxicity in each optical channel. The idea of ASEC (Application Specific Exposure Time) is to minimize the light dose first based on quantitative measurements and a decision support system and then to stabilize the signal over time in order to make the data analysis more dependable. Three different biological systems, the dis- and reassembly of the Golgi apparatus, autophagy dynamics and the mitochondrial membrane potentials were analyzed using ASEC to demonstrate the added value of this ARCO application. The autophagy dynamics experiment failed, but it is described in this thesis as a basic principle to investigate the effects of different light doses on biological systems.

In **Figure 42** the functional relation between the light exposure, information content and life time is illustrated as a rough approximation. Quite often only a small part of the image dynamic or more general information content is actually used for the read out of biological processes. Our hypothesis therefore was that ASEC can bring a clear added value to fluorescence live cell imaging by automatically identifying the most accurate exposure time for a specific application, and by stabilizing the signal over time. As it can also be seen in **Figure 42** the life time is directly related to the exposure time. The actual functional relation is not clear, and depends on many other experimental parameters besides the exposure time. However the experience has shown that both parameters are negatively correlated. The more light a cell received, the shorter its survival time becames.

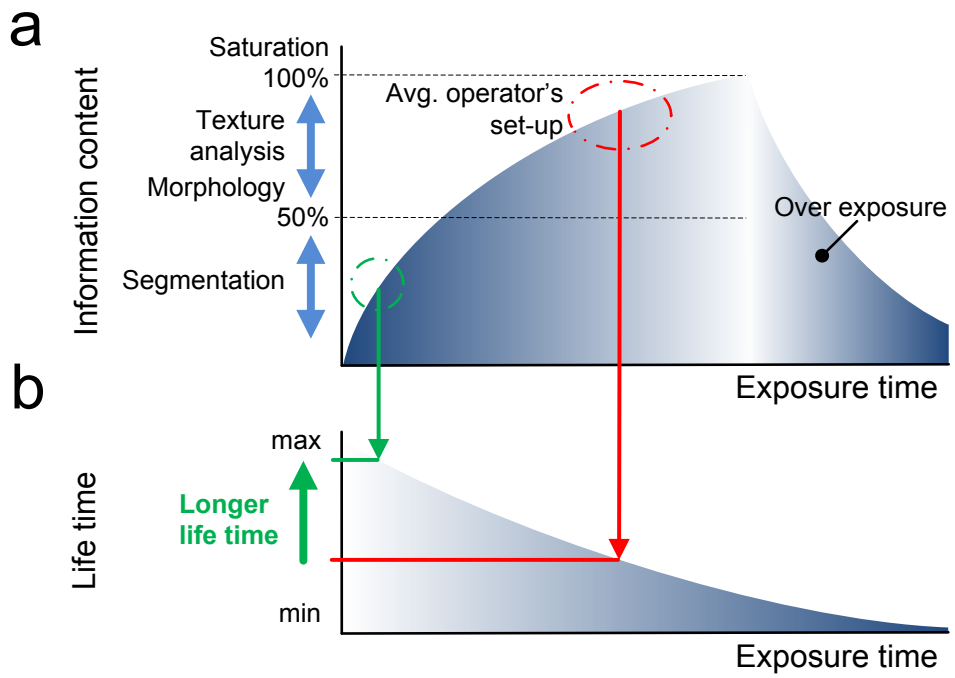


Figure 42 Relation between exposure time, information content and life time. (**a**) illustrates how the information content is related to the exposure and how this is approximately related to (**b**) the survival time [6]. Each dimension of an experiment can be put into this relation (such as sampling rate, magnification, resolution, focus- and XY-position), and the ARCO algorithm can be applied.

4.1.3 Materials

Biological samples

MCF7 nuclei are stained using DAPI and imaged with a 40x objective using an Olympus IX81 wide-field microscope. The sample is fixed. Note: all measurements are performed in real-time during the imaging workflow. A grid of 3x3 spots is measured, 4 spots with controlled light exposure, 5 spots without. Fig. 5 shows the workflow of the light exposure control system. Two measurements of a certain image feature are taken with two different exposure times. Based on this the functional relation of the exposure time and the chosen image feature are approximated. Finally the exposure time necessary to reach the target value of the image feature is calculated and set as the new exposure time. The parameter x is set to 0.25 (=> feature measured at +/-25% of the last exposure time).

Hardware, Software and Development Environment

The experimental setup consists of a motorized inverted microscope (Olympus IX81), a frame grabber (Matrox Meteor-II) and a CCD camera (Hamamatsu C9100-02). The motorized inverted microscope has a minimum focus step width of 0.01 µm. Two objectives were used, a 10x objective (Olympus UPlanFL, 3 µm depth of field, NA 0.3) and a 40x objective (Olympus LCPlanF1, 1 µm depth of field, NA 0.6). The microscope was operating in the brightfield mode. The focusing workflow we developed uses two different axial steps for different magnifications (Brazdilova and Kozubek 2009). Depending on the experiment, multi-point experiments were executed with a 20x objective. The 20x images were taken without binning. The automation of the image acquisition routine was implemented in C# and C++ using the native serial port commands of the specific hardware units. The methodology was integrated into the LifeXplorer framework.

4.1.4 Methods

Applying the general principle of ARCO to the dimension of light, the workflow is at follows. First, the operator's configuration of the exposure time is taken as a start value. As it is illustrated in **Figure 42**, the operator's guess for a suitable exposure time is often too high, because images with a brilliant contrast are only needed for image analysis which needs a very high signal to noise ratio. In the next step the exposure time is increased up to a level where the image gets saturated. In our case, we used a standard auto exposure function of the Nikon NIS-Elements software to find the upper border of the exposure time. This configuration is taken as a reference to evaluate all other configurations. The exposure time after that is decreased incrementally and the efficiency is calculated each time to classify the objects of interest, in our case mitochondrial membrane potentials or the Golgi apparatus. Finally the minimal exposure time is taken where still 95% classification efficiency is possible. 95% of the classification efficiency in the case of Golgi apparati means that 95% of the

area can still be segmented. In **Figure 43** (**a**) raw image with Golgi apparatus can be seen. The cells were exposed with 1 s exposure time. **Figure 43** (**b**) visualizes the segmentation efficiency for 100 ms exposure time.

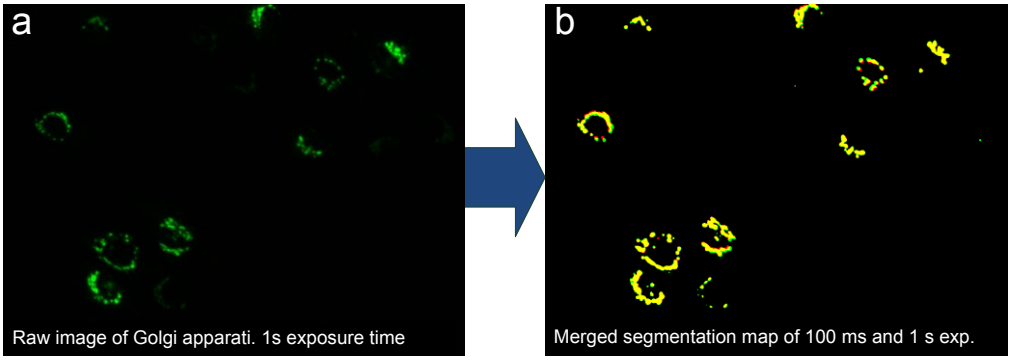


Figure 43 Golgi apparati segmentation efficiency for different exposure times. (**a**) shows a raw image of the golgi apparati taken with an exposure time of 1 s. (**b**) shows visualizes the segmentation efficiency if only 100 ms exposure time was used. The yellow areas are the overlapping areas of the segmentation maps created out of the raw images with 100 ms and 1 s exposure time. The red areas are missing areas when only 100 ms are taken.

The yellow areas are the overlapping areas of the segmentation maps created out of the raw images with 100 ms and 1 s exposure time. The red areas are the missing areas which appear if only 100 ms are taken. This demonstrates that 100 ms of light exposure is still enough to read out the fluorescence signal of Golgi apparati with the computer, although by human eye the image looks noisy and it's quality by that not suitable enough. 90% phototoxicity reduction however is gained.

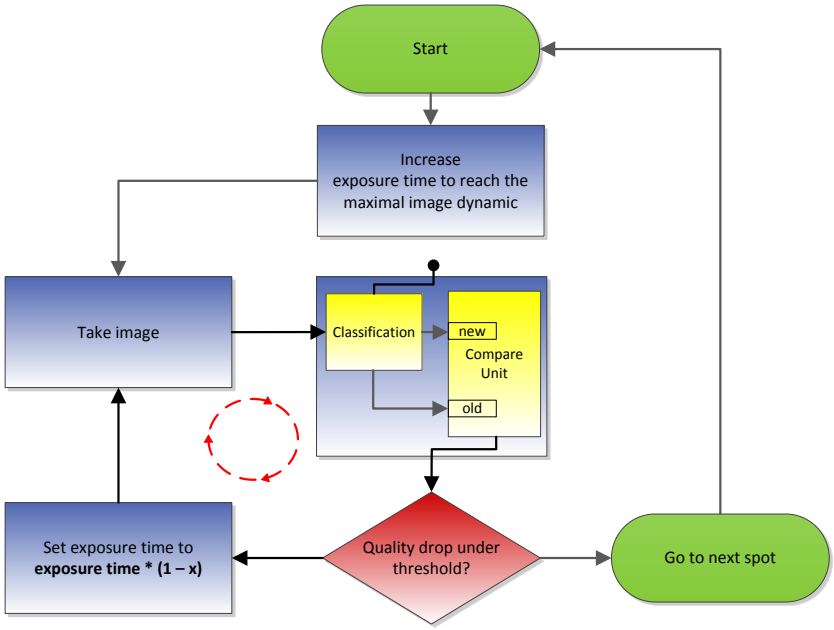


Figure 44 Signal minimizer. The exposure time is minimized based on a given, application-specific, processing unit. Afterwards the contrast with the minimized exposure time will be measured and will be kept constant over time.

The signal minimizer shown in **Figure 44** minimizes the exposure time based on a given, application-specific, processing unit. Afterwards the contrast with the minimized exposure time will be measured and will be kept constant over time. This method intends to keep not only the contrast constant, but rather the SNR, due to the fact that image processing often needs a constant SNR over time to work dependably. Because it is hard to define the SNR of an image, and in addition this value can be application-specific and to measure it can be highly defective. In addition it is not clear if the measured value has a linear functional relation to the exposure time, which causes even more conflict. A more general way to abstract from the SNR and to gain a linear relation to the light exposure is to measure a simple and dependable contrast value such as the

Absolute Gradient [117]. This algorithm sums up the absolute value of the first derivative:

$$F_{\text{abs grad}} = \sum_{\text{Height}} \sum_{\text{Width}} |i(x+1,y) - i(x,y)| \qquad \text{(Eq-11)}$$

This kind of contrast measure can also be considered an approximation of the "real" SNR, because with an increasing SNR, the contrast of an image will be increased approximately in a linear dependency.

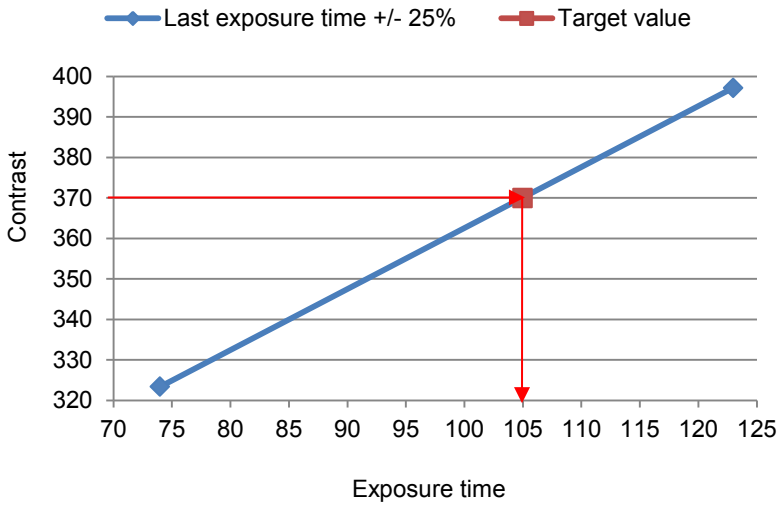


Figure 45 Approximates the necessary exposure time to reach the target value of a certain feature with a first order fit

Once the contrast quality, relatively to the first contrast value, drops below a certain threshold, the ASEC logic triggers the re-adjustment of the exposure time. **Figure 44** and Figure 46 illustrate the re-adjustment logic workflow. The principle how to re-adjust the initial exposure time such that the contrast stays stable is shown in **Figure 45**. Based on first order fit, the new exposure time is estimated, which is necessary to readjust the image quality such that the contrast quality stays equal to the reference value measured at the beginning of the screen. In order to perform a first order fit, next to the last measured value, two addition values need to be measured. Measuring two contrast values with two different exposure times, the new exposure time can be calculated by the following first order fit:

$$\mathrm{m} = \frac{c_1 - c_2}{t_1 - t_2} \qquad \text{(Eq-12)}$$

$$b = c_1 - m t_1 \qquad \text{(Eq-13)}$$

$$t_{exposure,corrected} = \frac{c_{target} - b}{m} \quad \text{(Eq-14)}$$

$$c \;\; contrast\ value, t \;\; exposure\ time$$

Originally the SNR was directly detected, but the results of the re-adjustment over time were not stable. The SNR is defined as

$$SNR = \sqrt{S - BG} \quad \text{(Eq-15)}$$

whereas S is the mean intensity of the foreground and BG the mean intensity of the background detected by a real-time segmentation. With the same quality control system that is presented in Figure 46 the system behavior was measured to stabilize the SNR.

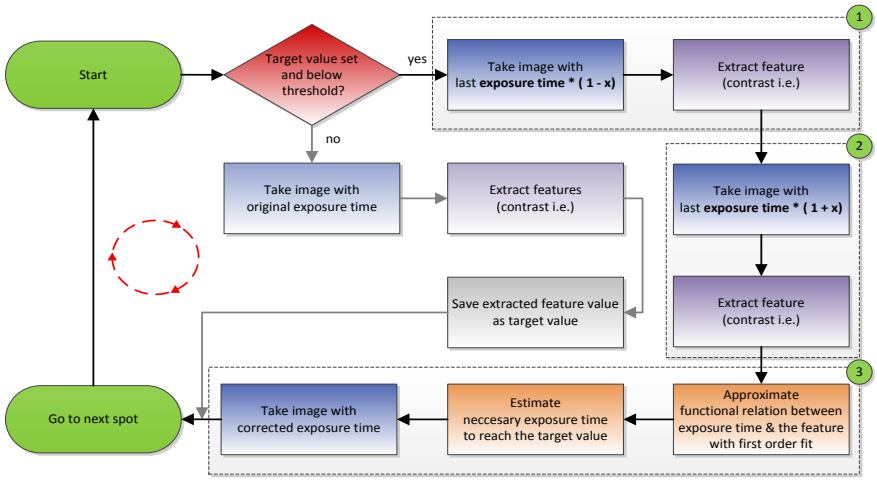


Figure 46 Controlled light exposure workflow. Two measurements of a certain image feature are taken with two different exposure times (parts (**1**),(**2**)). Based on this the functional relation of the exposure time and the chosen image feature is approximated. Finally the exposure time necessary to reach the target value of the image feature is calculated and set as the new exposure time (**3**).

4.1.5 Results

In **Figure 47** the development of the average intensity (average value over signal and background) with and without using the controlled light exposure method is illustrated. The signal of the non-controlled spots decreases by 15% on average after 220 imaging runs due to photo destruction and bleaching (fixed sample). The exposure time was set constant to 60 ms. In contrast, the signal decreases by only 1.5% using ASEC to control the light exposure over time. The controlled spots furthermore show a contrast precision error below 1% on average with no outliers in 220 imaging runs. Clearly it can be seen that spots with many cells (spot 07 with 90 cells) in contrast to spots with a lower amount of cells (spots 09 with 50 cells) show a stronger signal decrease over time. This effect appears due to the fact that the background can be considered as an offset for the signal and does not bleach with the same dynamics as the signal itself does. The background was not subtracted for the evaluation and this causes a variance in the bleaching curve of different spots.

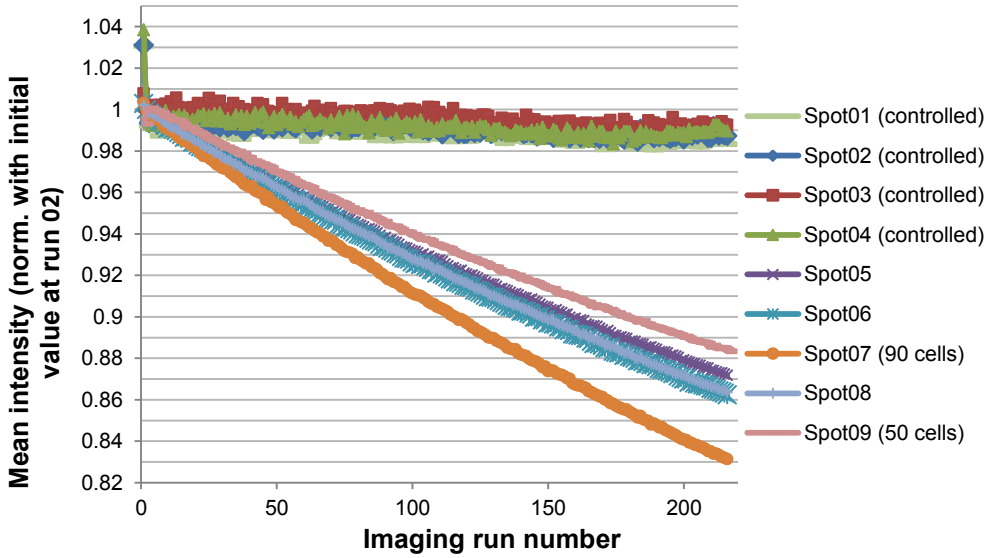


Figure 47 Average intensity development with and without using ASEC to control the light exposure over time

Figure 48 demonstrates that the exposure time has to be increased exponentially to keep the average intensity constant over time. The exposure time doubles after 90 images are taken. In each imaging run, however, three images are taken to re-adjust the exposure time constantly. Practically it is not necessary to re-adjust the exposure time in every run, for sampling rates which are higher than the frequency of the photobleaching dynamics.

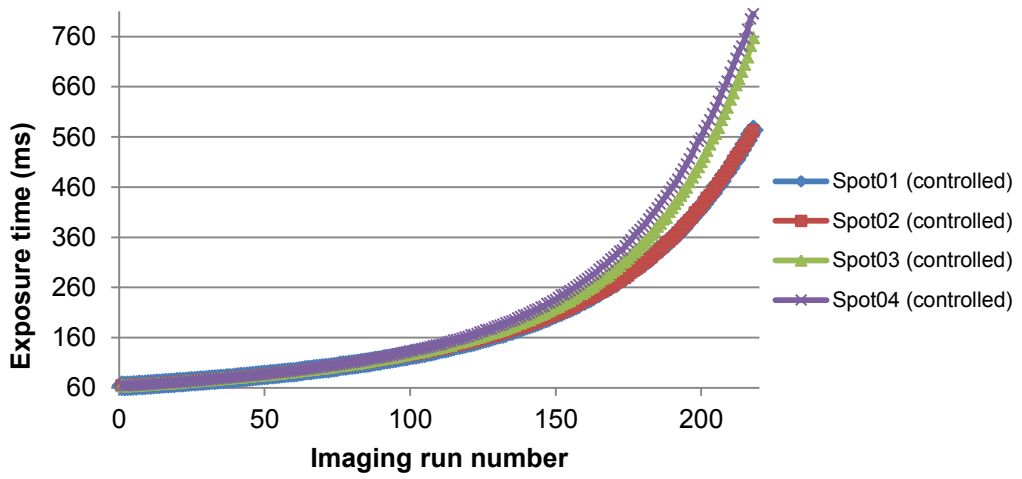


Figure 48 Exposure time development. Over time the exposure time has to be increased exponentially in order to keep the mean intensity constant.

ASEC with SNR control

It can be observed that the SNR stabilization works dependably until ca. 150 images are taken (**Figure 49**, **Figure 50**). After that the control system cannot keep the target value anymore. This happened because the functional relation was not calculated correctly, caused by the increasing non-linearity and steeper getting slope, which finally becomes exponential. Re-adjusting the parameter X by increasing it over time can solve this issue as well as taking historical data into account. Furthermore the measurements show that the SNR approximation is not dependable enough, because there are maximum turning points in the measurements of the non-controlled spots, where only minimum turning points should be (the light was switched on in the room). The decrease of the SNR is on average the same as the mean intensity decrease (85%), which is an indicator that the SNR calculation in principle works correctly. For the average intensity over time, all spots in **Figure 47** show a similar characteristic of the SNR development (**Figure 49**) due to the ROI (region of interest) detection that was performed in

advance of the screening to increase the cell count per image. As one can see the correlation of different experimental parameters such as XY-position and light exposure is significant once automated methods come into play. Classically, operators do not have the need to improve image quality on this precision level, because the image processing can be adapted manually for different quality cases.

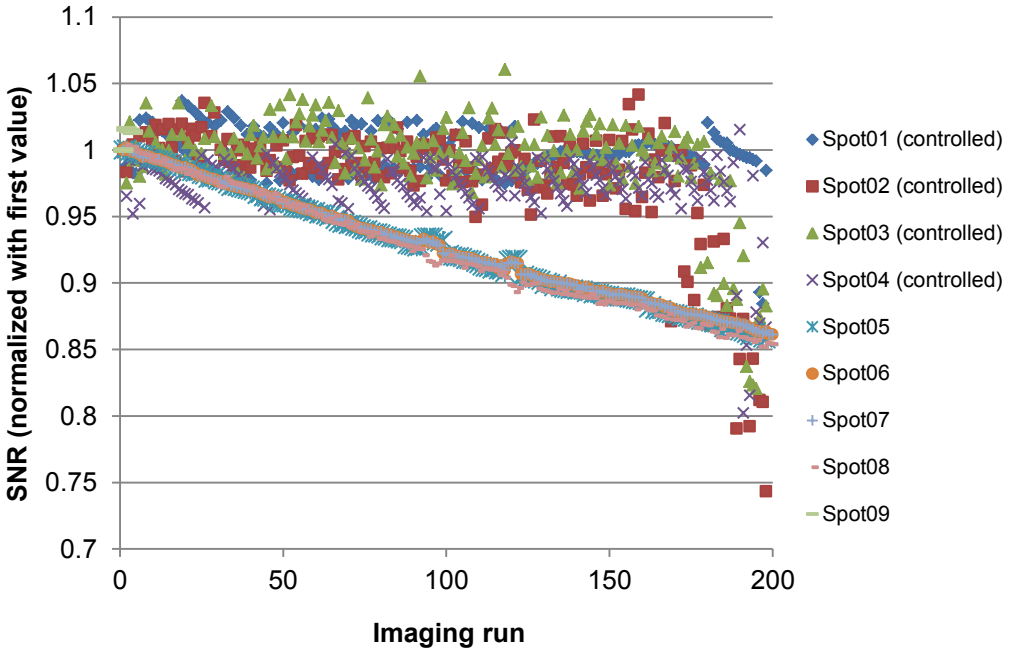


Figure 49 ASEC with SNR control

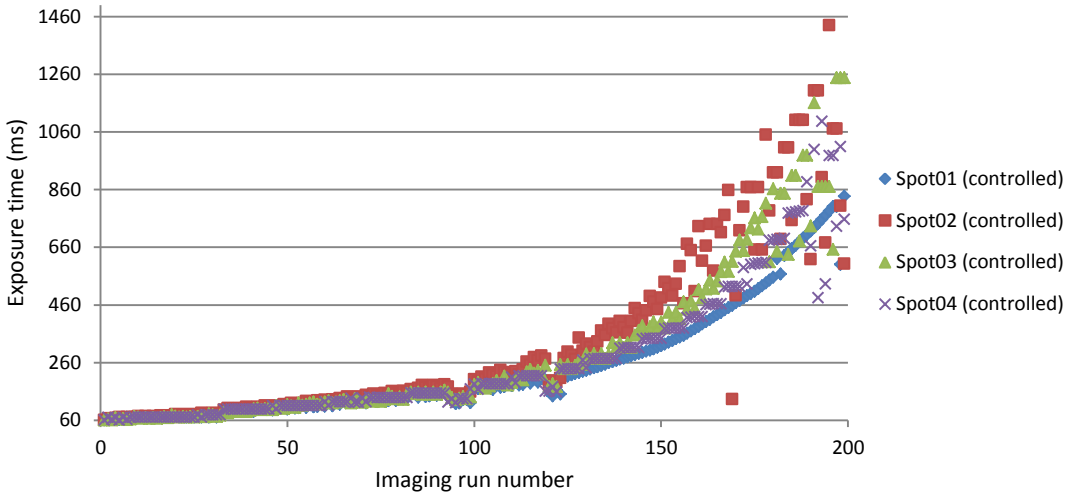


Figure 50 Exposure time development for ASEC with SNR control

Optimized time-lapse imaging of mitochondria energetics

Phototoxicity [5, 6] and photobleaching [7-9] are fundamental problems intrinsic to live-cell imaging, which reduce image quality, and it is widely recognized that light exposure results in mitochondrial dysfunction [33, 34]. To address this problem we performed experiments in HeLa cells loaded with TMRM (50 nM), imaging with the Olympus IX81 at 20x magnification for a period of 8 hours, at 30 second increments. ARCO was implemented to both optimize exposure time (ASEC) and optimize sample size through optimized XY-positioning of the measuring points (OSAPI). In addition the refocusing trigger (RFT) was applied to avoid unnecessary image acquisition and thereby to avoid unnecessary phototoxicity. Phototoxicity from TMRM excitation induces intracellular oxidative stress, propagating mitochondrial depolarization and mitochondrial ROS generation [33, 34, 43, 44]. In the absence of ARCO implementation, rapid loss of mitochondrial potential within the cellular population was detected between 1 and 2 hours of imaging. With ARCO-optimized light exposure, no sudden loss of mitochondrial membrane potential occurred, and we attribute the observed linear decrease over 8 hours (**Figure 51a+b**) to dye leakage and photobleaching. Thus at the time, ARCO achieved a 3 to 4-fold reduction in the impact of phototoxicity to 50% depolarization and total depolarization, respectively (**Figure 51b+c**). Furthermore, total intensity was significantly maintained using ASEC, demonstrating decreased photobleaching.

In contrast to the measurements mentioned above (**Figure 47-Figure 50**), the exposure time in this experiment was kept constant after minimizing it with the signal minimizer (**Figure 44**). Applying a refocusing trigger, the image contrast quality based on the focus position, however, was stabilized over time.

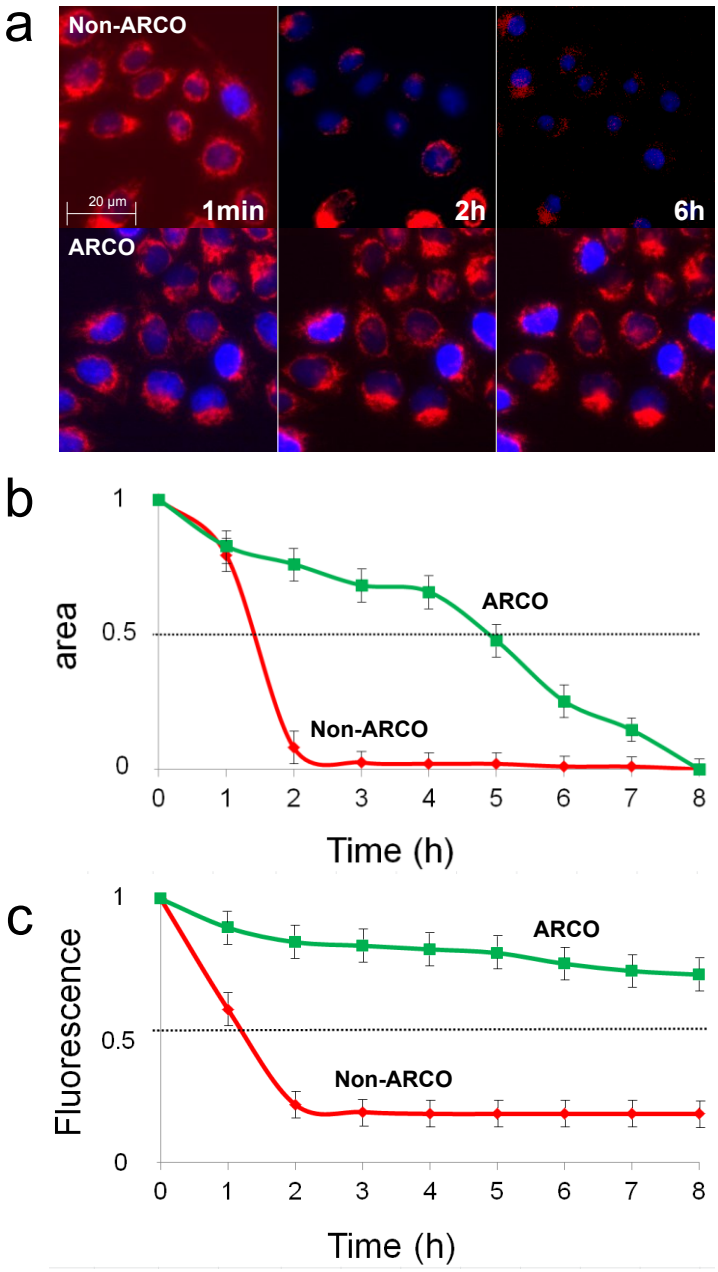


Figure 51 Mitochondrial energetics with and without applying ASEC

Optimized time-lapse imaging of Golgi reassembly

The Golgi apparatus is composed of membrane stacks near the nucleus, where proteins synthesized in the rough endoplasmic reticulum (ER) are sorted and modified before being secreted. Many enzymes are involved in this, and localize specifically to the Golgi compartment. The Golgi can be reconstituted from the ER demonstrating the intimate relation of the two compartments. How the Golgi is formed and maintained is of high interest, as it is intimately tied to vesicular protein transport [118]. Much of our knowledge is owed to the drug Brefeldin A (BFA), which leads to Golgi disassembly and relocation of Golgi markers into the ER [119]. Recently a live imaging approach has been used to quantitatively study Golgi assembly after BFA treatment, as a powerful tool to study regulation by overexpression and potential depletion of proteins [44]. Golgi reassembly following treatment with the protein transport inhibitor Brefeldin A (BFA) can be used to determine regulatory roles of specific proteins [120]. This approach requires the use of a fusion of ß-1,4-galactosyltransferase to cyan fluorescent protein (GalT-CFP) to detect Golgi reassembly over time based on quantifying juxtanuclear GalT–CFP localization following washout of BFA. Accurate quantification of Golgi reassembly requires that the process is not influenced by phototoxicity and that bleaching does not result in bias of measurement (e.g. underestimation of intensity). To that end, we applied ARCO in order to increase information content through shorter imaging time intervals during both disassembly in presence of BFA and reassembly after washout. Two illumination schemes were run in parallel. Exposure time was calculated by either (i) the standard "auto exposure" function present in all commercial imaging software packages or (ii) ASEC-based optimization of Golgi segmentation in an exposure series (**Figure 52a**). The auto-exposure function increased the exposure time from the initial selected 100 ms to 1 second, improving the signal-to-noise and image quality (Figure 4D, upper panel), but more importantly induced rapid bleaching of GalT–CFP (**Figure 52**c) and phototoxicity, as measured by rounding of the nucleus within 60 minutes, 30 minutes following BFA washout (**Figure 52b**).

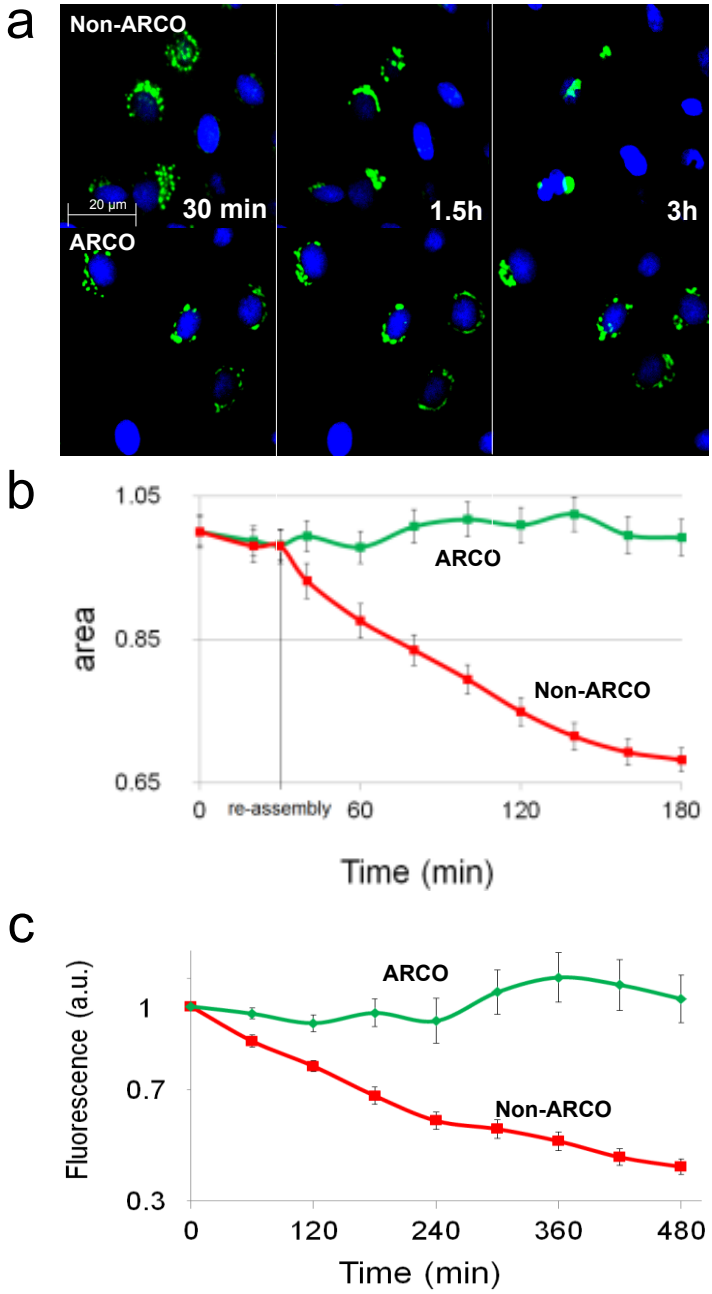


Figure 52 Golgi apparatus dis- and reassembly over time with and without applying ASEC

4.1.6 Possible integration to existing CLEM systems

Figure 53 illustrates the integration of the signal re-adjustment over time into the Controlled Light Exposure (CLEM) by Manders [10, 121]. In order to compensate the effect of the increasing exposure time (**Figure 50**) CLEM can be used. Additionally the initial exposure time can be minimized if a target value for the feature can be defined. I.e. for a given SNR which is sufficient to segment cells for a cell counter, the exposure time can be minimized. The workflow for the signal minimization is described in **Figure 44**. The costs for additional measurements of the re-adjustment can be reduced if the re-adjustment factor for the exposure time is calculated for one spot and is taken for all others spots within the same well, assuming that spots of the same experiment have the same bleaching dynamics and similar cell counts.

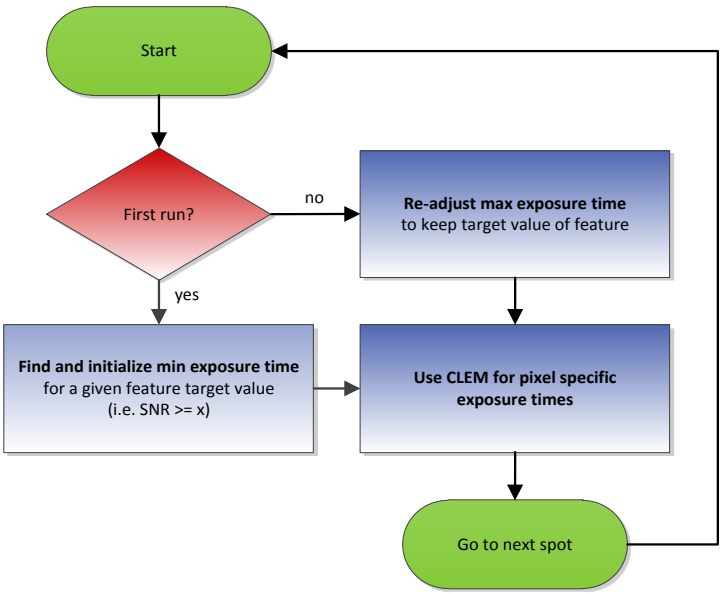


Figure 53 Possible integration into existing Controlled Light Exposure (CLEM) systems

4.2 Autophagy dynamics with ASEC

4.2.1 Autophagy dynamics

Autophagy or autophagocytosis is a catabolic process in cell biology which mainly involves the degradation of cell components by the lysosomal machinery. The process plays a crucial part in the relevant processes like cell development, growth and homeostasis. In the process the cell breaks down its own components or foreign particles and recycles them as it can be seen in **Figure 54**. Autophagy is a precisely regulated process by the cell and aimed to maintain the balance between recycling, synthesis and degradation of cellular products.

"The autophagy machinery is thought to have evolved as a stress response that allows unicellular eukaryotic organisms to survive during harsh conditions, probably by regulating energy homeostasis and/or by protein and organelle quality control. The same machinery might therefore be expected to diversify functionally in complex metazoan organisms, so as to regulate new layers of defences used by multicellular organisms to confront different forms of stress. A plethora of genetic, biochemistry, cell biology, systems biology and genomic studies have recently converged to support this notion. The autophagy machinery interfaces with most cellular stress-response pathways [122], including those involved in controlling immune responses and inflammation. This interface is not only at the level of the autophagy pathway, but also entails direct interactions between autophagy proteins and immune signalling molecules[123]." [124].

4.2.2 Material

Biological samples

To assess the impact of phototoxicity on aggregation of autophagosomes, stably transfected GFP-LC3 MCF7 cells were plated in ibidi 8-well microscopy

slides (ibidi, Munich, Germany) at 50000 cells per well the day before treatment. Prior to analysis cells were washed with PBS and put in HBSS solution (containing Penicillin/Streptomycin and Hepes buffer), and immediately imaged. Addition of the lysosomal inhibitor Bafilomycin A1 (final concentration 100nM) to HBSS was used as a positive control for aggregation of autophagosomes.

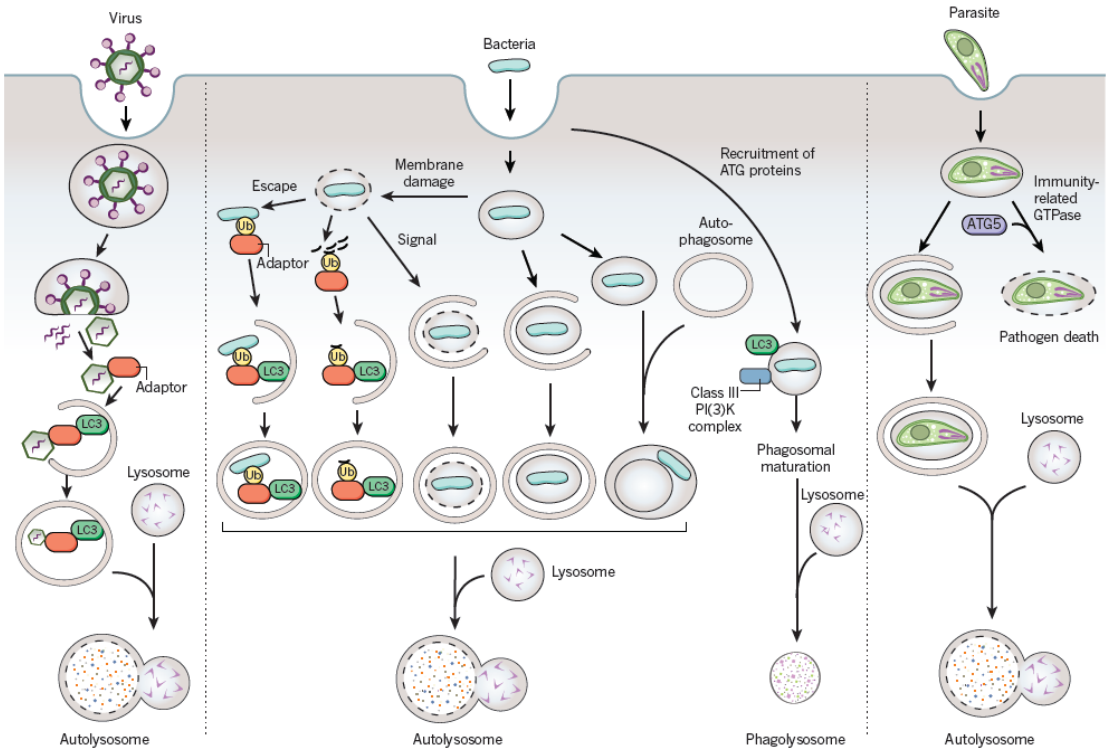


Figure 54 "Possible autophagy-protein-dependent pathways of pathogen degradation. Possible pathways involving the autophagy machinery by which viruses, bacteria (and damaged membranes of bacteria-containing vacuoles) and parasites may be targeted to the lysosome. Adaptor refers to the proteins shown in the cargo-recognition box in Fig. 1; however, as yet undiscovered adaptors may be involved in pathogen recognition, and pathogen targeting may involve ubiquitin-dependent or -independent mechanisms." [125] Image adapted from [125]

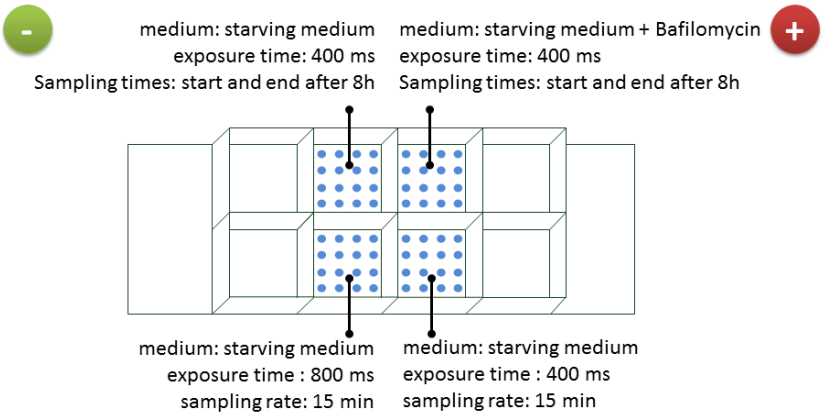


Figure 55 Autophagosomes / LC3 development over time (7.5h) under different exposure times. Four experiments, each in its own well with 25 measuring points

Hardware, Software and Development Environment

The experimental setup consists of a motorized inverted microscope (Olympus IX81), a frame grabber (Matrox Meteor-II) and a CCD camera (Hamamatsu C9100-02). The motorized inverted microscope has a minimum focus step width of 0.01 µm. Two objectives were used, a 10x objective (Olympus UPlanFL, 3 µm depth of field, NA 0.3) and a 40x objective (Olympus LCPlanF1, 1 µm depth of field, NA 0.6). The microscope was observing in the brightfield mode. The focusing workflow we used uses two different axial steps for different magnifications [1]. 400 sampling points were measured with 5 µm axial steps using a 10x objective and 1 µm axial steps using a 40x objective. The 40x images were taken with 2x binning and 60ms exposure time. For lateral sampling a 10x10 regular grid with an accuracy of 900 µm was taken in each chamber. Three wells were imaged using binning and one well was imaged without binning. The evaluation of the non-binned images showed that the evaluated algorithms can be heavily irritated by noise. Our presented evaluation therefore only considers binned images. The automation of the image acquisition routine was implemented in C# and C++ using the native serial port commands of the specific hardware units. The surface modeling was implemented using Mathematica and focusing algorithm ranking done with Matlab. Both were integrated using the distributed microscopy framework LifeXplorer.

4.2.3 Methods

Aiming to find a light sensitive biological system, autophagy as a crucial process of cell recycling was tested against different light doses. The scientific question first of all was if there is any effect on the degradation of autophagosomes influenced by light due to apoptotic and autophagic responses [122, 126]. In order to get an answer for this qualitative question, the autophagosomes marker LC3 was measured in four experiments as specified in **Figure 55**. In two different experiments, the cells were imaged with a low and a high exposure time. And in another two experiments a positive and negative control was measured. All experiments were imaged simultaneously. All experiments were run as multi point experiments with 16 sampling points in separated well for each experiments to get robust results. In all cases, starving medium as taken in order to put the cells under stress to see autophagy dynamics more easily [127]. The first experiment was image cell with 400 ms light exposure and the second to observe them with 800 ms, with a sampling rate of 15 min over 8 hours. As positive and negative controls another two experiments were run in parallel. Each well for both control experiments was only imaged twice, at the beginning and at the end of the experiment. The positive control was treated with Bafilomycin which blocks the degradation of autophagosomes [128]. The negative control was only starving medium without any treatment of the cells. Both control experiments were measured with 400 ms exposure time. The hypothesis was that high light exposure should bring the cell status closer to the positive control and autophagosomes should aggregate over time as it can be seen in **Figure 56**.

4.2.4 Results

The result we were interested in was a binary answer to our question, if the imaging itself changes the biological process and is therefore light sensitive. Our hypothesis was that high light exposures over time finally lead to cell death and

therefore accelerate the process of autophagy. The result however was that the system was not light sensitive. By eye, in some cases an aggregation of the Lc3 particles could be observed as it would have been necessary in order to come closer to the positive control of BFA that inhibits the degradation of autophagosomes. In **Figure 56a** one sampling position is shown over time. The cell marked with the red circle beautifully shows the effects that we thought will be raised by higher light exposures due to phototoxicity and cell death effects. However we could not reproduce the effect that higher light exposures cause the cells to behave as in the case of the positive control. On average the cells behaved almost the same. The phenotypes of the cells did not vary clearly if they were sampled or not. **Figure 56b** illustrates the quantitative results of your investigations. The red curve is the positive control with BFA blocking the degradation of autophagosomes. The green line is the negative control. It is equal to the average behavior of the cells if they are not treated with any drug influencing the process of autophagy. Both control wells were only sampled at the beginning and at the end of the experiment. The blue curve is equal to the lower light exposure with 400ms, whereas the yellow curve shows the quantitative results of the high exposure. In contrast to what we could have expected, both curves are close to the negative control. From this it can be derived that the imaging itself did not significantly inhibit the degradation of light as in the case of the positive control. In contrast higher light exposure in these experiments sometimes, as it gets obvious in **Figure 56b**, even seemed to cause cellular homeostasis. In other experiments the effect was different and the higher light exposures and the imaging caused the aggregation of Lc3.

In all cases the effect was not reproducible yet and only single cells were affected, not the whole cell population. We therefore stopped using this system as a possible candidate for light sensitive biological systems and investigated two system that are well known to be sensitive to oxidative stress and thus phototoxicity causing the creation of reactive oxygen species [33, 34, 43, 44, 112, 113]: mitochondria [33, 43, 129] and the Golgi complex [120, 130, 131].

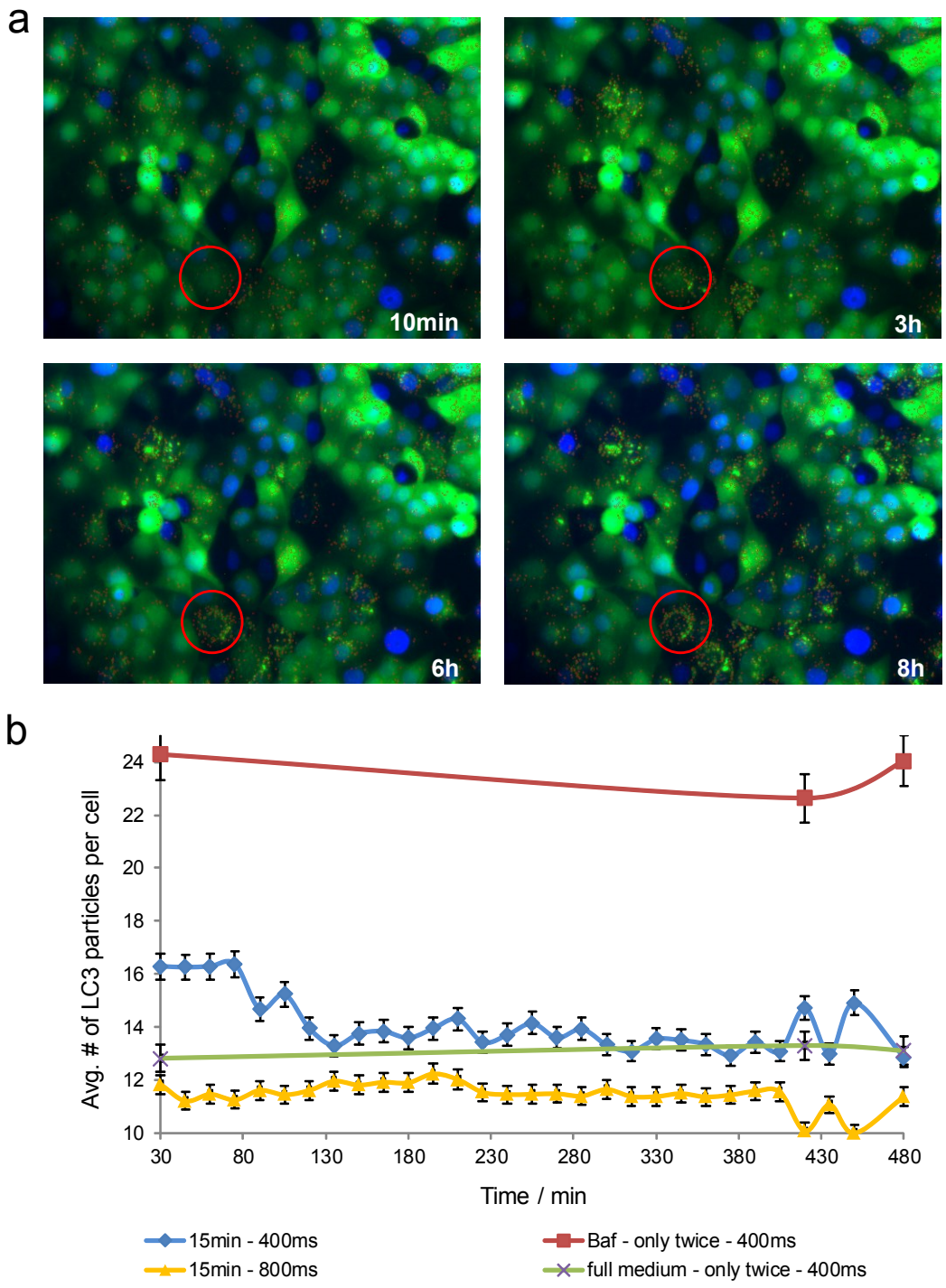


Figure 56 Lc3 particles over time with different exposure times. No significant effect of higher light exposures could be observed.

4.3 Mitochondrial energetics with ASEC

4.3.1 Mitochondria

"Mitochondria are central to the process of programmed cell death that kills damaged or superfluous cells. Surprisingly, components of the death machinery turn out to be essential for keeping these organelles in shape." [132]. They are the ATP generation, calcium homeostasis, and integrate metabolic signaling complexes of cells. TMRM is a lipophilic, positively charged and accumulates within mitochondria matrix according to the Nernst equation due to the mitochondrial membrane potential across the inner mitochondrial membrane (-180 mV). Upon depolarization, TMRM is rapidly released to cytosol and medium. Importantly, phototoxicity from TMRM excitation can induce endogenous response within the cell, resulting in waves of mitochondrial depolarization and mitochondrial ROS generation [33, 34, 43, 129, 133]. The use of mitochondrial membrane potential reporters is fundamental for cell death and toxicity readouts. However, at present, no strategy has been demonstrated for empirically minimizing the imaging effect, i.e. optimizing long-term monitoring of mitochondrial function. **Figure 56** illustrates important cell components including a single mitochondrion.

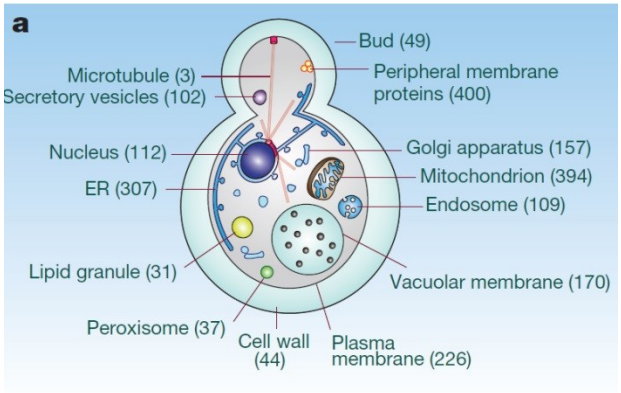


Figure 57 Yeast membrane compartments. Numbers indicate membrane proteins (MPs). (Babu, Vlasblom et al. 2012)

4.3.2 Material

Biological samples

Hela cells were plated at 40000 cells per well in ibibi microscopy slides one day prior to the experiment. For analysis of mitochondrial depolarization, cells were stained with Tetramethyl Rhodamine Methyl Ester (TMRM) at a final concentration of 10nM for 30 minutes at 37°C, 5% CO2. Cell nuclei were visualized through DNA staining with Hoechst 53342, at a final concentration of 1µg/µl, incubation at 37°C, 5% CO2 for 30 minutes. Live cell imaging was performed in full medium (DMEM, 10%FBS).

Hardware, Software and Development Environment

The experimental setup consists of a motorized inverted microscope (Olympus IX81), a frame grabber (Matrox Meteor-II) and a CCD camera (Hamamatsu C9100-02). The motorized inverted microscope has a minimum focus step width of 0.01 µm. Two objectives were used, a 10x objective (Olympus UPlanFL, 3 µm depth of field, NA 0.3) and a 40x objective (Olympus LCPlanF1, 1 µm depth of field, NA 0.6). The microscope was observing in the brightfield mode. The focusing workflow employs two different axial steps for different magnifications [1]. 400 sampling points were measured with 5 µm axial steps using a 10x objective and 1 µm axial steps using a 40x objective. The 40x images were taken with 2x binning and 60ms exposure time. For lateral sampling a 10x10 regular grid with an accuracy of 900 µm was taken in each chamber. The evaluation of the non-binned images showed that the evaluated algorithms can be heavily irritated by noise. Our presented evaluation therefore only considers binned images. The automation of the image acquisition routine was implemented in C# and C++ using the native serial port commands of the specific hardware units. The surface modeling was implemented using Mathematica and focusing algorithm ranking done with Matlab. Both were integrated using the distributed microscopy framework LifeXplorer.

4.3.3 Methods

As shown in the chapter above, where Golgi complexes were segmented using different exposure times, ASEC was also applied to mitochondria systems. Two parameters were measured as a readout for phototoxicity and photobleaching. The total size of the mitochondria was measured to approximate the active amount of mitochondria as an indicator for effect of phototoxicity. To measure the photobleaching, the mean brightness of the mitochondria was measured over time as an approximation for the fluorescence intensity. The imaging was performed with a sampling rate of 30 seconds over eight hours. The data analysis was performed based on single cells and then accumulated again to a mean value over time. To finally evaluate the added value of ASEC, the phenotypes of the cells were classified visually in order to compare healthy and unhealthy cells, and the temporal distance between both phenotypes. Four independent experiments were performed, to ensure the effects presented here are statistically significant, and thus in principle reproducible. Each experiment was based on a special imaging workflow. Two sampling points were measured with ASEC and the refocusing trigger logic activated. ASEC calculated 10 ms to be accurate enough to detect ≥ 95% of the mitochondria. Another two positions were measured with the normal exposure time of 40 ms and with permanent refocusing. The refocusing trigger, as is explained in greater detail in the corresponding chapter, optimizes the refocusing behavior of the microscope such that the refocusing step is only performed if the image quality drops under a configured threshold, here set to 90%. Unnecessary imaging is thus avoided based on the ARCO algorithm, which calculates whether the result / costs ratio is higher with or without the refocusing step. The image quality result is considered to become 100% again if the refocusing step is done. And at the same time the results function is set to one for all value ≥ 90%. This means that the ratio only gets worse if the quality of the image drops below 90%, which then triggers a refocusing based on the result / costs optimization paradigm. In order to have a reference to compare both experiments with, four control positions were imaged only at the start, and after eight hours.

4.3.4 Results

Figure 58a illustrates what the cells looked like after measuring them with and without any imaging optimization. After six hours the mitochondria could hardly be detected anymore. The brightness however is enhanced so that it is possible to see the fluorescence signals with the human eye, and thus identify the remaining mitochondrial structures. After only two hours, the phenotypes clearly show that most of the mitochondria are not functioning anymore, which causes a switching off of the fluorescence marker, and on average an incremental dimming of the signal. In contrast the phenotypes of the cells which were imaged with ASEC and the refocusing trigger look equivalent to the reference cells, which were only imaged at the start and the end of the experiment. The refocusing trigger, as can be seen in the corresponding chapter, only necessitated refocusing the measuring positions on average every hour. In addition, an average of six images had to be taken to refocus a position again. The phototoxicity which is caused by the refocusing was reduced approximately to 1%. **Figure 58b** shows the quantitative result of the average mitochondria size measured over time. After only 1.5 hours the size of active mitochondria has dropped to 50% for the Non-ASEC measurements. In contrast, the ASEC and RFT measurements show a clear trend and drop to 50% of the total mitochondria size after five hours. When taking the 50% as a reference threshold to evaluate the phototoxicity, cells can be measured 3.33 fold longer using ASEC and RFT. Taken 10% as the lowest threshold for mitochondria activity, cells can be measured four times longer. The loss of fluorescence signal (in a.u., arbitrary units) is only 33% (25% loss instead of 75%) as strong as in the case of normal imaging.

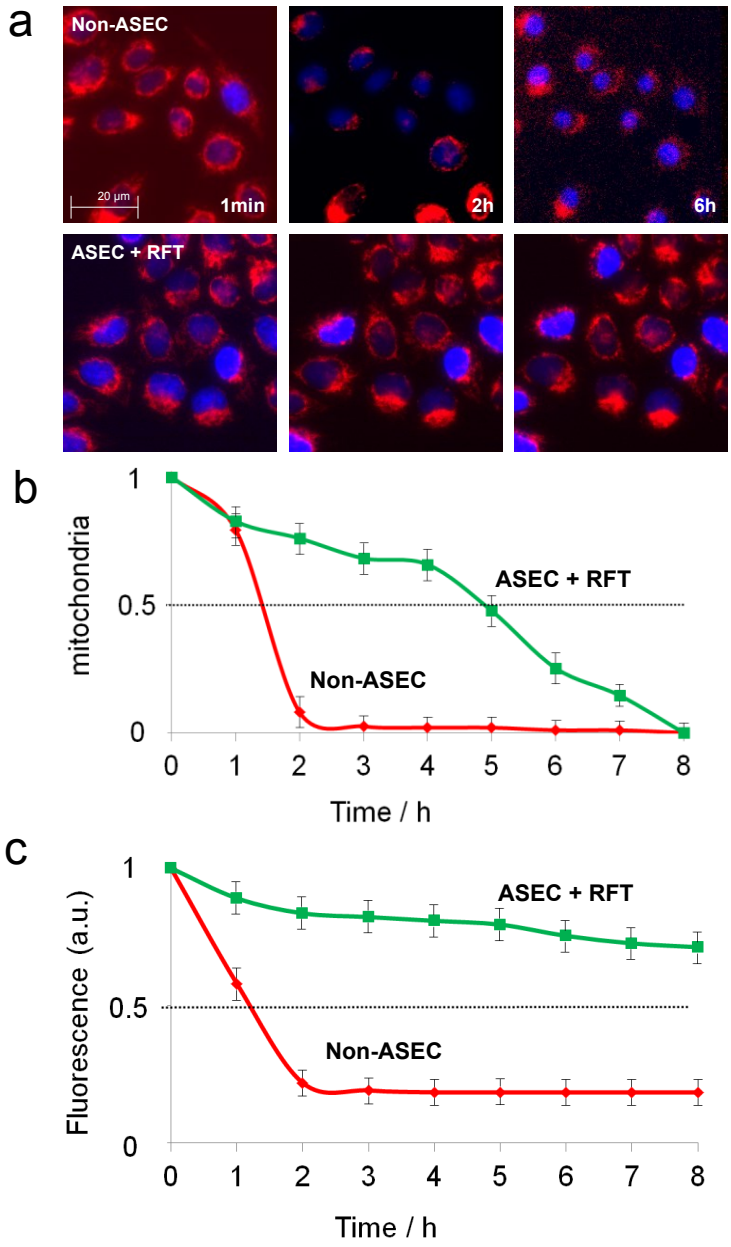


Figure 58 Phototoxcity and photobleaching evaluation of ASEC using the mitochondria as a light sensitive biological system.

4.4 Golgi reassembly with ASEC

4.4.1 Golgi dis- and reassembly

The Golgi apparatus is an organelle structure devoted to process the proteins, which are synthesized in the endoplasmic reticulum (ER). It is also known as the Golgi complex, Golgi body, or simply the Golgi, and was found in 1897 by the Italian physician Camillo Golgi [134, 135]. The Golgi reassembly is well known to be influenced by phototoxic stress [130].

"Reactive oxygen species (ROS)/reactive nitrogen species (RNS) and ROS/RNS-mediated oxidative stress have well-established roles in many physiological and pathological processes and are associated with the pathogenesis of many diseases, such as hypertension, ischemia/reperfusion injury, diabetes mellitus, atherosclerosis, stroke, cancer, and neurodegenerative disorders. It is generally accepted that mitochondria play an essential role in oxidative stress because they are responsible for the primary generation of superoxide radicals. Little attention, however, has been paid to the importance of the Golgi apparatus (GA) in this process." [130]. **Figure 59** illustrates the assembly and disassembly of the Golgi between the ER and the plasma membrane (PM).

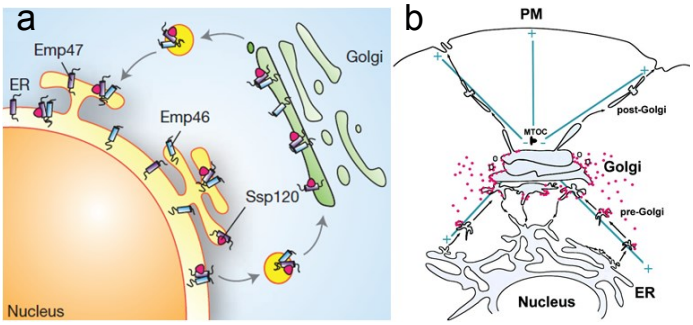


Figure 59 (**a**) Model showing recycling of the Emp46–Emp47–Ssp120 complex between endoplasmic reticulum (ER) and early Golgi [136]. (**b**) Diagram of the Secretory Pathway Transport intermediates in the form of vesicles and tubulovesicular intermediates mediate forward and retrograde transport between the ER, Golgi complex, and plasma membrane (PM). [137]

4.4.2 Material

Biological samples

NRK cells stably expressing GalT-CFP (NRK-GalT-CFP) were a gift from the Starkuviene lab and cultured as described [43]. Cells were plated in ibidi 8 well microscopy slide one day prior to imaging. The GalT-CFP redistribution assay was performed as described [4]. In short, cells were stained with Hoechst 53342 for 15 minutes prior to acquisition and after multiple positions per well were defined in the software, 5 µg/mL of BFA and 0.1 mg/mL of cycloheximide were added. Timelapse images were acquired every 2 minutes over 30 min. After washing cells 3 times with the preheated growth medium, time-lapse imaging of the same positions was resumed for 3 hours. Cycloheximide was kept in all solution during Golgi reassembly to prevent de-novo synthesis of GalT-CFP.

Hardware

The microscopy setups consisted of the motorized inverted microscope Nikon Ti-E, run in wide field fluorescence. The Nikon Ti-E with an automated table and perfect focus system operated with Nis-Elements AR 4.1, using a 20x Plan Fluor objective (Nikon, NA 0.75) and an interline transfer CCD camera (Clara, Andor). The metalhalide illumination (Nikon Intensilight) was attenuated to 25%, rather than using low exposure times, which might then be shorter than the reaction time of the epifluorescence shutter (Sutter 10-2, Sutter Instruments), gating exposure. Initial exposure was estimated by using the auto exposure function based on saturation of pixels. This value was reduced by 40% to avoid potential saturation of pixels at other XY-positions. Microscopy computers were connected with the LifeXplorer software through network access for online image analysis.

4.4.3 Methods

Due to the fact that phototoxicity causes the creation of ROS [6, 34, 138] and the fact that the Golgi assembly is influenced by ROS, the hypothesis was that

imaging itself would negatively influence the process of the dis- and reassembly over time. More concretely the Golgi-to-ER relocalization of GalT-CFP was analysed with high and low light exposures after brefeldin A (BFA) addition and wash-out. Adding BFA makes the cells disassemble their Golgis, whereas after washing it out, the Golgis reassemble. The hypothesis was that light exposures that are too high will disturb the cells dis- and reassemble their Golgis. As a reference for the high light exposure, the auto exposure function of the Nikon NIS Elements Software was used to determine the exposure time which operators would use in order to get the highest dynamic range of their images. The result was 2.5 seconds. Assuming that biologists might know that this exposure time will be to high concerning the viability of cells and the throughput of the microscope, the high exposure time was set to 1 second. ASEC was used then used to determine the lowest exposure time possible to classify Golgis. A macro was plugged into the graphical interface for the operator that communicates with the LifeXplorer framework. Once this basic macro triggers the ASEC logic hosted in the LifeXplorer framework, new macros for the microscope are written automatically. These macros extend the basic macro, which waits until the workflow extensions are programmed by the ASEC logic. The workflow as described above basically decreased the light exposure given on a configured decrement step. For these experiment series, the configuration was 100ms, which means that a series of images is taken between 1 second and 100 ms light exposure. Afterwards ASEC uses the signal minimizer to identify which classification result based on the lowest exposure time possible still is suitable to reach 95% segmentation quality. The reference segmentation map is the one created out of the highest exposure time, since this image has the highest SNR and the highest dynamic range. As can be seen in **Figure 60**, the Golgi apparati are captured with the high and low exposure, classified, segmented, after which the resulting segmentation maps are cross-correlated. **Figure 60d** illustrates the resulting merged map. Yellow pixels are classified for both exposure configurations, whereas red pixels are lost in the case of the low exposure time. Using the human eye however, the missing 5% of the segmented Golgi area are not relevant, which also holds true for the statistical analysis. The focus plane of 16 sampling positions was sampled each two minutes using ASEC and the same

configuration of another 16 positions were imaged without using ASEC. Initially the cells were treated with BFA to start the disassembly of the Golgi complexes and after 30 min BFA was washed out to make the cells reassemble the Golgis. Before and after running these two experiments, reference positions in separated wells were taken to compare the low and how exposure times with the phenotypes of cells that were not imaged at all. In addition, another experiment was run where the same amount of spots with and without using ASEC cells was imaged over 8 hours without any BFA treatment.

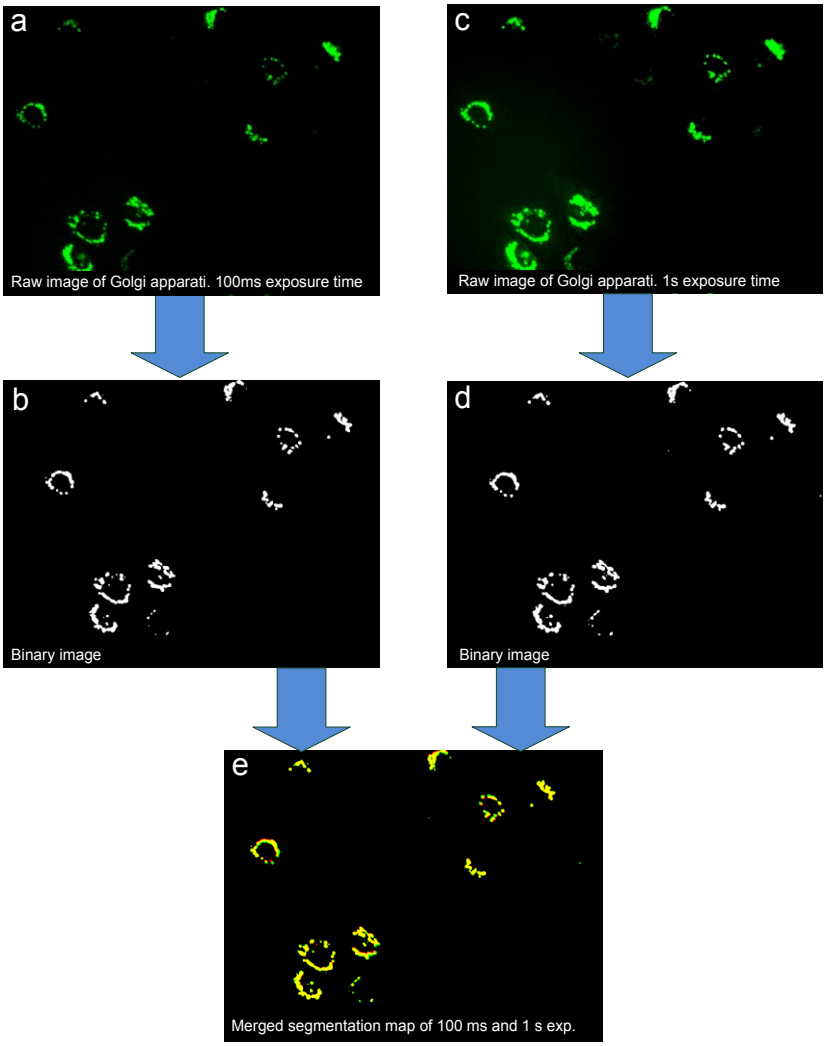


Figure 60 Cross correlation of two different segmentation maps as a quantitative measure for the classification efficiency at low (**a**) - (**b**) and high (**c**) – (**d**) light exposures. (**e**) 95% of the segmented are can be segmented with the low exposure time, which is equivalent to the yellow pixels. Only 5% of the Golgi area is not segmented using a low exposure time of 100 ms. 90% of the light exposure and thus phototoxicity however is saved.

4.4.4 Results and Discussion

Both experiments, with ASEC and Non-ASEC adjusted light exposure, were analyzed based on a single cell population wide data analysis. Two parameters then were aggregated and plotted over time. The first parameter was the nuclei size over time. **Figure 61a** clearly visualizes that over time the cells detach from the surface, and the nuclei become smaller and rounder in the case of the high light exposure times. **Figure 61b** clearly shows that the nuclei size over time stays stable using ASEC, whereas after ca. 30 minutes of imaging the cells which were imaged without using ASEC start to suffer from photobleaching and shrinking cell nuclei. **Figure 61a** shows the plotted, quantitative effect of **Figure 61b**. **Figure 61c** in addition measures the fluorescence as the second parameter over time (a.u., arbitrary units). Again, in the case of ASEC the Fluorescence stays stable and cells even go into mitosis, disassembling and reassembling their Golgi complexes. That cells reproduce in a normal way is an obvious and steady criteria that the phototoxicity using ASEC was reduced down to a level where cells are not negatively influenced in their production and reproduction. This poses a contrast to the cells imaged without using ASEC. They clearly show that on average the photobleaching effect heavily influences the readout by lowered intensities over time. In numbers the phototoxicity of ASEC in this experiment could be reduced at least six-fold, considering that the nuclei size did not change at all after 180 min using ASEC, whereas with Non-ASEC the cells behaved abnormally after only 30min. In the case of the photobleaching measure, the photobleaching could be reduced 100%.

These results make clear that experimental optimizations using the ARCO algorithm increase the change of stable results. Certainly ARCO, as well as any other ARCO derivate, needs to be specifically and carefully implemented for each individual experiment application. The benefit from this however is that experimental setups are mathematically more robust and thus the reproducibility can be increased, besides the benefits of lower phototoxicity and information densification. Commonly, no biologist or image processing expert would in the default case test crucial experimental parameters on a quantitative basis. In

between the ability to image live cells in the first place, and reproducibly image and evaluate live cell system is a big gap, which today is partially closed by the experience of individuals. The access threshold for non-experts however is still too high for biologists to be able to fully independently perform quantitative experiments. Yet in the future, complexity can only be handled based on reliable ways of automating science. ASEC on the dimension of light exemplarily demonstrates the effects of a targeted automation approach such as the ARCO algorithm. ASEC allows biologists to automatically optimize the data quality based on their specific scientific question and biological phenotypes.

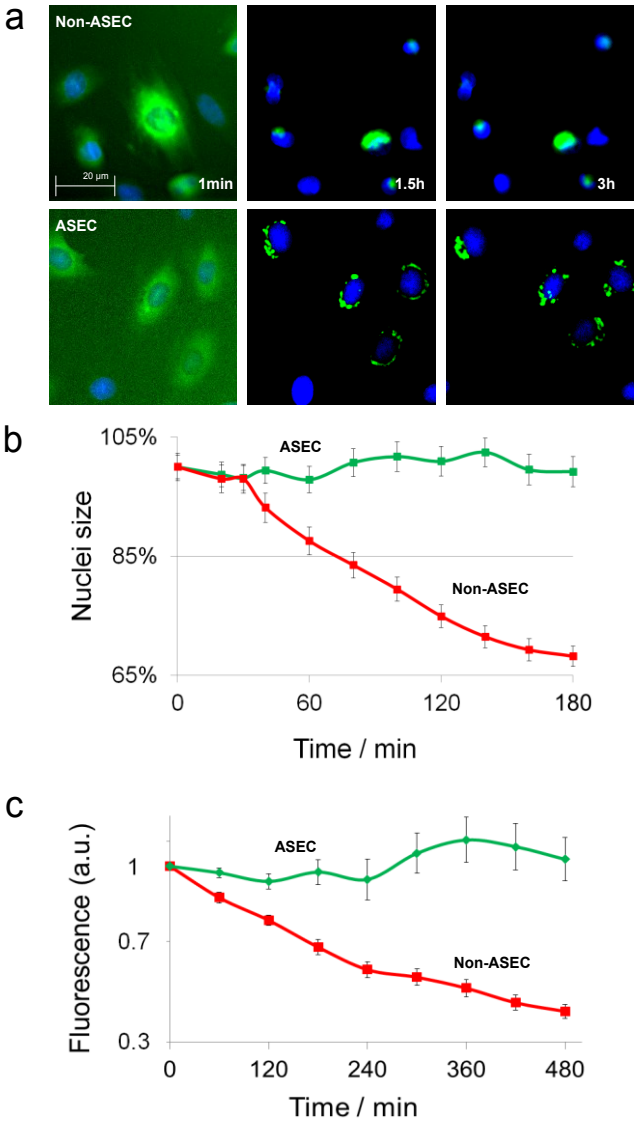


Figure 61 Phototoxicity and photobleaching evaluation of ASEC using the Golgi complex as a light-sensitive biological system.

4.5 MBAAS (Model Based Autofocusing Algorithm Selection)

4.5.1 Abstract

Crucially, the significance of the analysis output is a function of the initially acquired raw data, and therefore strictly controlled in order to minimize error propagation. In high-throughput microscopy the balance between image quality and acquisition rate is a fundamental optimization step. The auto focusing step represents a relevant target for increasing the throughput of microscopes and the sampling rate for live cell screens. Improved focusing methods have the potential to both minimize exposure bleaching and phototoxicity, and increase sampling speed. The author proposes an adaptable, data-driven algorithm based on the ARCO algorithm, which improves data quality by increasing the amount of in-focus images, minimizes exposure times, and is applicable to high throughput microscopy. The algorithm provides an automatic selection of the best focusing algorithm based on the modeling of the slide surface. Our method in addition leads to higher speed and image quality by maximizing the amount of in-focus images while reducing the effort to find the in-focal planes. The autofocusing could be accelerated three- to fourfold while the number of in-focus images could be doubled.

4.5.2 Introduction

Rapid high-resolution imaging is emerging as an essential tool in quantitative biology. Since data analysis depends on the incoming data, the data quality of the raw images in light microscopy is highly relevant for the effectiveness in gaining insights into biology. In high-throughput microscopy a conflicting goal is to provide fast imaging on one side, and to retain the quality of the data acquisition on the other side. One of the most important prerequisites for high-quality data is precise focusing. The accuracy of focusing algorithms differs with different data and therefore the choice of the best algorithm depends on the experiment. This can be seen in literature [2, 3, 139, 140], where the evaluation of focusing algorithms showed different results if the data was different. Within one image optimized, estimations of the focal planes of single sub areas can be computed. Image stacks can be cut into sub areas, and for each of these areas the focus position can be calculated separately [141]. In addition this has shown that the focus position is related to the physical unevenness. In order to speed up the focusing step, a sub sampling of the observed area can be used [141]. Several patterns, i.e. a regular grid or more sophisticated ones using Halton sampling points can be used to increase speed. In addition, different adaptive algorithms with variable z-steps can be used to accelerate the focusing step [1]. Pre-processing the acquired raw data before computing the in-focus plane was analyzed as suitable to generalize the usage of focusing algorithms [142].

Using the same focusing algorithm for different experiments, the number of in-focus images can greatly vary. So far, the main reference to evaluate performance of focusing algorithms for a certain type of data was done by manual inspection (i.e. [3]). In order to maximize the amount of in-focus images, however, it would be optimal to automatically select the best focusing algorithm before starting a high-throughput experiment. To that end the author developed a method which makes it possible to select the best focusing algorithms for a specific experiment automatically. A model of the slide surface was used as a reference to automatically evaluate autofocus algorithms. Outliers furthermore can be detected automatically. After having found the best algorithm for a

certain type of experimental data, the autofocus step can be accelerated up three- to fourfold.

17 algorithms have been evaluated, out of which 12 were stable against noise and are presented in the results section. Using the surface information about local gradients, outliers were detected on the fly. Our approach helped to better understand systematic errors of algorithms as can be found in existing literature [3]. Using our method, the most suitable focusing algorithm can be found automatically. The best selected algorithm has been applied to a relevant biological sample to quantify the possible acceleration of the data acquisition.

4.5.3 Materials and Methods

Focusing procedure

The focusing process consists of a basic idea. The Z position of the objective is changed with a given strategy and a certain quality of the image is measured. In principle, operators would like to see sharp images, which means that the autofocus algorithm will for example measure a certain contrast value of the image. It will then try to find the maximum quality possible and assume that the in-focus plane is at this position. The relevant literature [1-3, 139, 140] presents and evaluates different quality measurements. These optimizations focus on the amount of in-focus images, which means that the algorithms should be as close as possible a manual measure of the in-focus plane. In addition, strategies have been presented for the z-sampling [143, 144] to reduce the amount of necessary sampling steps. All existing evaluations however are data- and thus experiment-related. For each experiment, the results of a selected algorithm or strategy can be completely different.

Overview of the approach

The approach presented here therefore takes existing focusing methods and evaluates them, automatically selecting the most suitable algorithm for a given experiment dataset. The workflow of the approach is illustrated in **Figure 65**. In the first run, the first well is taken as a reference well to select the most suitable

focusing algorithm that maximizes the amount of in-focus images, and afterwards the lowest subsampling to sample the surface and approximate non-sampled z-positions. The number of in-focus images here is the result function for ARCO, whereas the computing time is the costs function. 100 spots are taken in a 10x10 grid, and for each focusing algorithm a low frequency, third order model is created out of the XYZ-point cloud, using Mathematica. The highest number of in-focus images is then approximated using the standard deviation of the error function. The error can be estimated with the differences of the measured and the approximated z positions of surface model:

$$s^2 = \frac{1}{n-1}\sum_{i=1}^{n}(\Delta z_i - \Delta z_{mean}) \qquad \text{(Eq-16)}$$

The algorithm with the lowest variance can be considered as being the most suitable algorithm to estimate the focus positions. This focusing algorithm is then selected to process the most efficient sampling rate to sample the surface and find the focus plane. The 10x10 regular grid which was taken to sample the surface, is then sub sampled incrementally until only four corner points are taken. For each sub sampling step, the surface is modeled again, and it is calculated how many images the focusing would still need to find the focus plane with the approximated values of the surface. By reducing the sampling rate of the surface, the precision of the approximation gets incrementally lower. I.e. a first order fit cannot approximate local distortions of a slide's surface, which means that the error of the model is incrementally increased by decreasing the sampling rate. At the same time only few sampling points have to be taken in depth, which reduces the costs to create a model. The sampling points are taken with a fixed volume to be stable against the slide distortion independently from the distance of each sampling point. These measurements are therefore static for the amount of images taken, whereas all other positions are taken with a dynamic focusing method using a dynamic extreme search, [143] and have to rely on the first estimate of the focus position by using the modeled surface.

17 algorithms were evaluated, out of which 12 could be taken for further evaluations plus one which was developed for special purposes. These 13 algorithms are presented here.

The algorithms are:

1. Absolute Gradient
2. Brenner Gradient
3. *Edge Gradient*
4. Energy Laplace
5. Genetic Programming
6. Laplacian
7. Laplacian Variance
8. Manuel measure
9. Netten Filter
10. Sobel-Tenengrad Variance
11. Squared Gradient
12. Tenengrad
13. Vollath F4

The appendix provides further information about the algorithms. The Edge Gradient is a new algorithm, which was developed for experiments on autophagy dynamics [124, 145, 146] with a high-resolution imaging, especially using a 40x magnification objective. **Figure 62** illustrates how measured values of a focus position are composed. The first component is the surface offset. Due to uneven surfaces this offset very likely will vary in each XY position. On top the biological offset has to be considered. This offset finally determines where the in-focus position is, because next to the surface offset that is physically unique, the fluorescence markers in each channel can be located in different planes.

Unfortunately in reality the position will be flawed. The standard deviation of the error therefore has to be considered as the key figure that determines the precision of the focusing algorithm. To automatically select the best focusing algorithm, a reference is needed in order to rank them. The reference so far has been a manual measure.

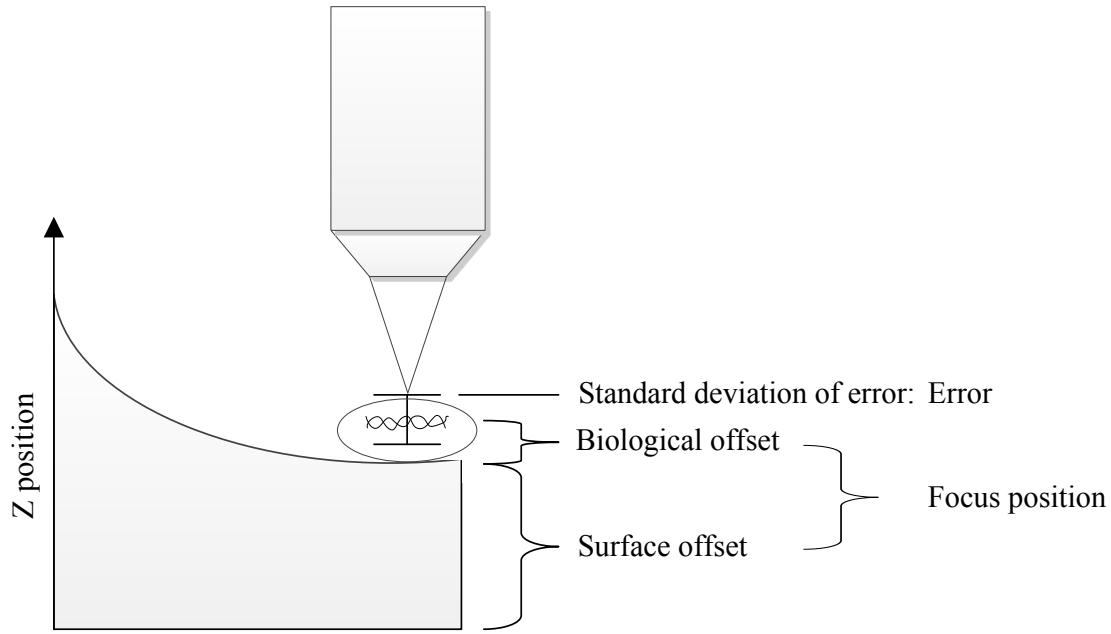


Figure 62 Components of a measured value.

Our hypothesis was that a low order fit of the measured focus positions represents the slide surface and is suitable as a reference. For better understanding, the model for the measured value is illustrated in **Figure 62** and described as follows:

$$m(x,y) = s(x,y) + b(x,y) + e(x,y) \qquad \text{(Eq-17)}$$

m *measured value*; s *surface offset*; b *biological offset*; e *meaurement error*

The error e itself consists of two parts:

$$e(x,y) = a + p(x,y) \qquad \text{(Eq-18)}$$
$$a \;\; accuracy;\; p \;\; precision$$

By fitting a measured point cloud of XY- and Z-positions, the high frequency function components p is filtered out and e gets close the a. If the accuracy a is close to zero, only p has to be computed in order to find the best focus algorithm. The precision of an algorithm is equal to:

$$p(x,y) = f(x,y) - m(x,y) \qquad \text{(Eq-19)}$$
$$p \;\; precision; f \;\; fitted\ value;\; m \;\; meaured\ value;$$

The standard deviation of all precision values finally is a suitable measure to compare the precisions of algorithms with the same accuracy. This chapter evaluates the automatic selection of the focusing algorithms and compares it to the manual selection. The surface model was taken as the ground truth. The calculated focal planes were compared to the focal plane of the manual measure. In addition, the author shows how the surface model can be used to detect outliers, as well as to accelerate the focusing step by sub-sampling.

Hardware, Software and Development Environment

The experimental setup consists of a motorized inverted microscope (Olympus IX81), a frame grabber (Matrox Meteor-II) and a CCD camera (Hamamatsu C9100-02). The motorized inverted microscope has a minimum focus step width of 0.01 µm. Two objectives were used, a 10x objective (Olympus UPlanFL, 3 µm depth of field, NA 0.3) and a 40x objective (Olympus LCPlanF1, 1 µm depth of field, NA 0.6). The microscope was observing in the brightfield mode. The focusing workflow used employs two different axial steps for different magnifications [1]. 400 sampling points were measured with 5 µm axial steps

using a 10x objective and 1 µm axial steps using a 40x objective. The 40x images were taken with 2x binning and 60ms exposure time. For lateral sampling a 10x10 regular grid with an accuracy of 900 µm was taken in each chamber. Three wells were imaged using binning and one well was imaged without binning. **Figure 63** illustrates all four wells and the topological map of the sampled surfaces. The evaluation of the non-binned images showed that the evaluated algorithms can be heavily irritated by noise. Our presented evaluation therefore only considers binned images. The automation of the image acquisition routine was implemented in C# and C++ using the native serial port commands of the specific hardware units. The surface modeling was implemented using Mathematica and focusing algorithm ranking done with Matlab. Both were integrated using the distributed microscopy framework LifeXplorer.

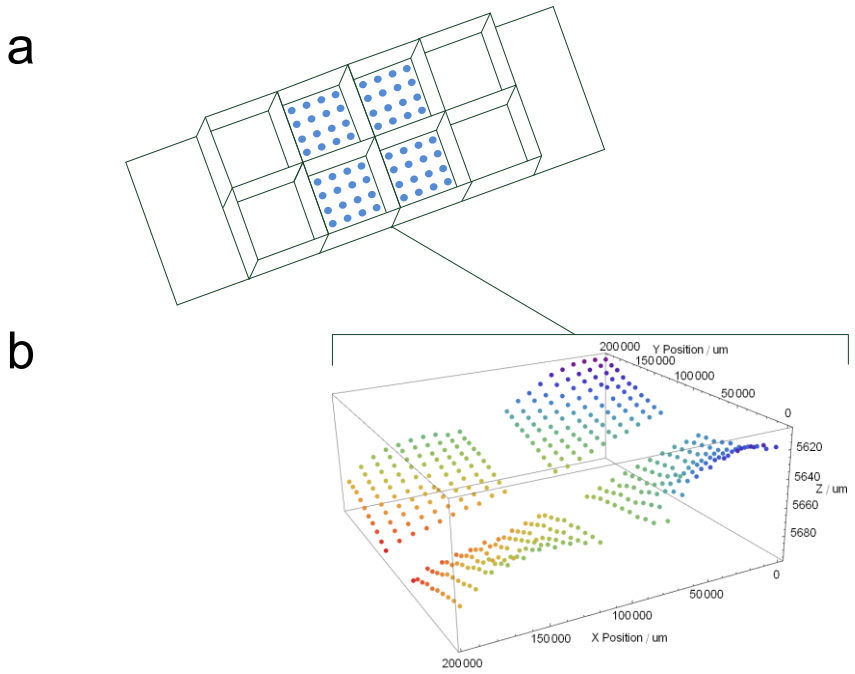


Figure 63 (**a**) Schematic representation of the microscopic preparation used in this study. Four wells of an iBidi slide were sampled, each with a 10x10 regular grid pattern. The 400 colored points represent the z positions for each XY position. (**b**) All positions were analyzed with 17 focus algorithms, out of which 12 algorithms were found suitable for the automatic selection of the best focus algorithm method.

Biological samples

Human MCF7 breast cancer cell lines (Cell Line Services, Heidelberg) were maintained in DMEM/10% FBS/L-glutamine/non-essential amino acids/penicillin/streptomycin/amphotericin B (Invitrogen). Cells were plated in 8-well microscopy µ-slides (iBidi) at a density of 20,000 cells per well. To label nuclei, cells were incubated for 15 min with Hoechst 33342 at a concentration of 1 ug/mL at 37C. Cells were then fixed with 4% paraformaldehyde (Electron Microscopy Services) in PBS, and stored in PBS (Sigma).

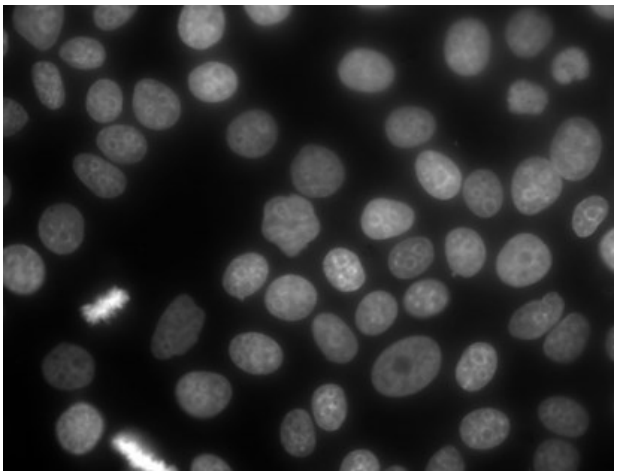


Figure 64 Cell nuclei of MCF7 cells.

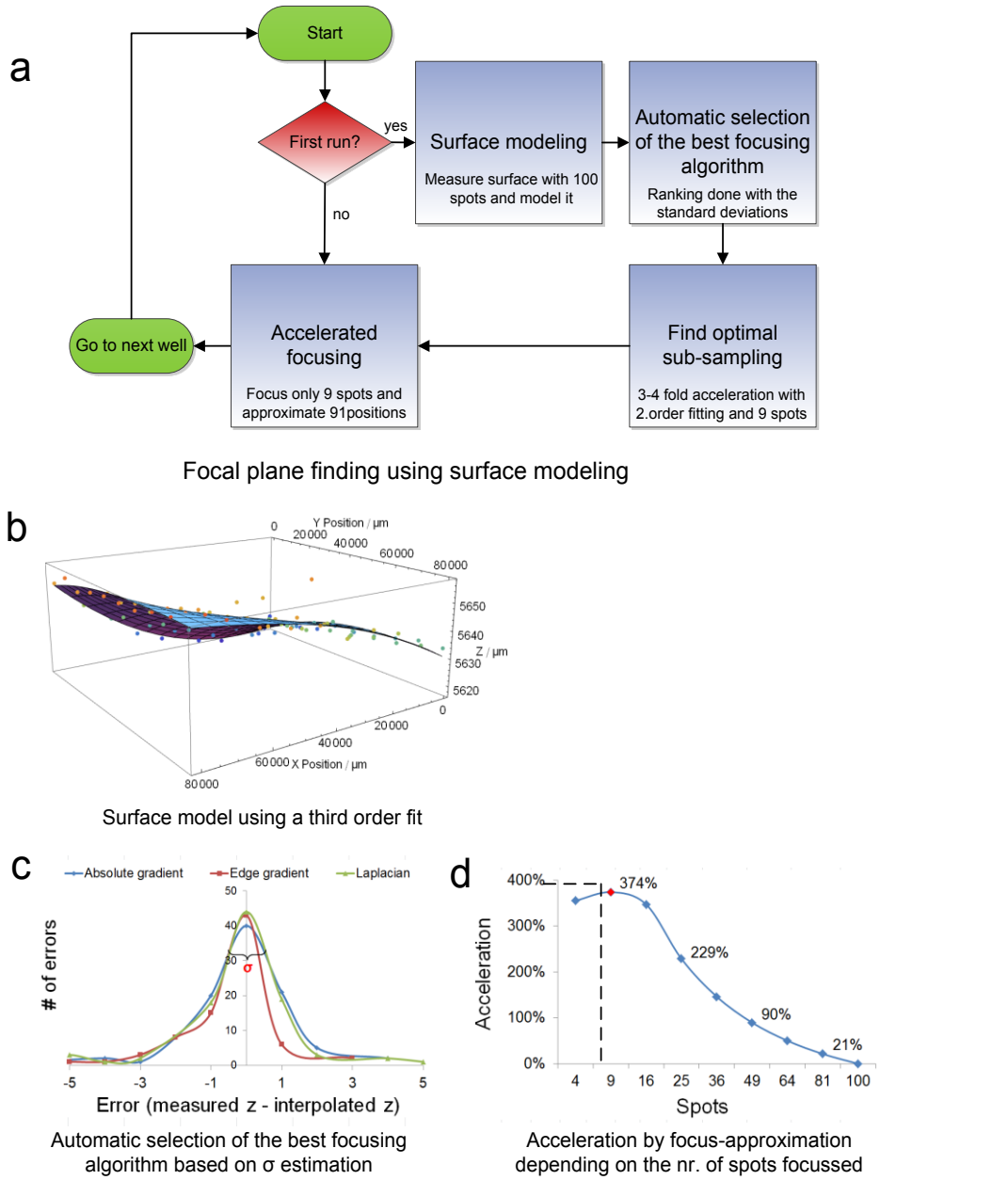


Figure 65 Surface modeling an acceleration workflow. Automated selection of the best focus algorithm and acceleration of the autofocus step. Part (**a**) shows how to accelerate the focusing step while the maximal amount of in-focus images is reached. Sampling a reference well of the slide with many positions, a surface model can be created. The model serves as a ground truth reference, which can be used to automatically select the best focus algorithm. In a next step, the optimal sub-sampling for all other wells can be computed, which maximizes the focusing while the maximal amount of in-focus images is kept. (**b**) illustrates a result of the surface modeling using a third order fit. (**c**) exemplarily visualizes three different algorithms with the same accuracy, but different precisions. Part (**d**) shows an acceleration curve depending on the number of focused and approximated spots.

Focal plane finding using surface modeling to minimize algorithmic errors

Automatically finding the real in-focal planes out of defective measures involves a method to minimize the error of the measured focus positions. To that end, the author presents an approach that uses surface approximation to calculate the in-focus positions. To understand the method, it is crucial to take a detailed look at the structure of the measured focus positions. The model for a measured focus position can be found in **Eq-17**. The focus position consists of a physical offset of the surface plus the biological offset, which depends on where the parts of the specimen are located, which were stained, and are observed with the aid of the fluorescence signal. The error amplitude of normally distributed focusing algorithms is equal to the standard deviation of the error (**Figure 62**). The higher the standard deviation is, the lower the precision of the focusing algorithm will be. The aim is reduce the precision error to close to zero in order to get the real in-focus positions. The low frequency part of the measured focus positions is the bowing of the slide surface, whereas the high frequency part is the error component of the focusing algorithm. Assuming that normally distributed focusing algorithms are evaluated, the mean error is equal to the accuracy of the algorithm. If the algorithm's accuracy is comparable to the accuracy (mean error) of the manual measure, the surface fit can be considered as a ground truth reference. **Figure 65b** illustrates a third order fitting of 100 sampled spots of a 1cm² well using the Edge Gradient algorithm to get the in-focus plane. To automatically select the best focusing algorithm for particular data, the author evaluated a first, second and third order polynomial to model the surfaces of iBidis. The distortion of the surface in addition showed that a second order polynomial cannot be sufficient, whereas a forth order fitting led to over fitting of the data. By fitting the measured point cloud with a third order plane, the low frequency part of the measured values can be modeled efficiently.

Automatic selection of the best focusing algorithm based on error estimation

From the above, a suitable reference was created to calculate the precision of all algorithms. The surface model of each algorithm can be taken to calculate the precision of all algorithms individually. The standard deviation of all differences between the focus position of the surface model and the measured focus position as shown in **Figure 62** is equal to the precision of an algorithm. Our hypothesis was that the lower the standard deviation is, the higher is the precision of the focusing algorithm, and the higher the amount of the in-focus images becomes. To prove that this holds true for our test data, both rankings of the precision measures and the manual, by eye evaluation were compared. The results can be found in **Table 2**. The manual evaluation in addition to the precision measure contains the amount of images which are in-focus by eye as a visual reference. The criteria which were evaluated are described again as follows.

1. Number of in-focus images

The number of in-focus images is related to the manual measure with a 1 µm tolerance, which means that one plane before or after the manual measure is still counted as in-focus considering the fact that the manual measure is also defective.

2. Precision

This precision is equal the standard deviation of all differences between the approximated in-focus positions and the measured ones.

3. Accuracy

The accuracy is the mean value of all differences between the focus position of the surface model of the manual measures and the focus position of the focusing algorithms.

4. Ranking

Three different rankings have been calculated based on the criteria above to get an impression of the correlation between the rankings.

Table 2 Automatic and manual overall ranking of 12 focusing algorithms in comparison

	Surface reconstruction as reference		Manual reference			
Focusing algorithm	Precision (StdDev)	Ranking with surface reconstruction	Nr of in-focus images (+/- 1µm) in %	Ranking with in-focus images	Accuracy	Ranking with accuracy
Manuel measure	1.09	(3)	100%	-	0.00	-
Edge Gradient	0.94	2	91.67	1	0.63	10
Absolute Gradient	0.87	1	89.33	2	0.79	11
Laplacian	1.11	3	85.00	3	0.52	9
Laplacian Variance	1.12	4	81.67	4	0.18	2
Squared Gradient	1.34	6	82.67	5	0.40	7
Vollath F4	1.32	5	82.00	6	0.25	4
Netten Filter	1.44	8	81.33	7	0.39	5
Brenner Gradient	1.41	7	81.33	8	0.23	3
Tenengrad	1.48	9	75.00	9	0.11	1
Energy Laplace	1.61	11	76.67	10	0.48	8
Sobel Tenegrad Variance	1.59	10	73.00	11	0.39	6
Genetic Programming	1.92	12	45.00	12	1.27	12

Table 3 Precision statistics for focusing algorithms for all wells

	Precisions / Standard deviations			
Focusing algorithm	Well02	Well03	Well04	Avg.
Edge Gradient	1.08	0.78	0.96	0.94
Absolute Gradient	1.05	0.72	0.83	0.87
Manuel measure	1.11	0.99	1.17	1.09
Laplacian	1.36	1.03	0.94	1.11
Laplacian Variance	1.22	0.96	1.18	1.12
Squared Gradient	1.28	1.43	1.30	1.34
Vollath F4	1.37	1.30	1.28	1.32
Netten Filter	1.61	1.43	1.27	1.44
Brenner Gradient	1.53	1.44	1.27	1.41
Tenengrad	1.43	1.53	1.49	1.48
Energy Laplace	1.55	1.65	1.63	1.61
Sobel Tenegrad Variance	1.50	1.67	1.59	1.59
Genetic Programming	1.70	1.85	2.21	1.92

Accuracy and focusing algorithm classification

To use the automatic selection of the best focusing algorithm, it has to be assumed that the accuracy of the analyzed algorithms is close to zero. In this case it is possible to rank the focusing algorithms based on their precision only. It was proven that 12 of the 17 algorithms evaluated in this chapter can be related to the manual measure and are normally distributed. Others however showed a biased error (> 1 µm depth of view) from which it can be derived that the offset is a systematic error, and from there that the biased algorithms measure a different quantity than our eye does. **Figure 66** illustrates the effect of bias algorithms. In order to automatically select the best algorithm, the algorithms have to be classified due to the fact that the surface reconstruction will by design integrate any systematically existing offset, and put the fitted plane in the middle of all measured focus positions.

To automatically classify which algorithms built a common accuracy class, a histogram of all errors of all algorithms was created. The surface modeling did not work out for all algorithms with non-binned images, binning therefore was activated. **Figure 66** illustrates the histogram for 100 sampling points, which were taken in one well. As it can be seen here there are two different algorithm classes. **Table 4** breaks the accuracy classes down to the algorithms which create them. Both classes and their algorithms belong of normally distributed focusing algorithms. For non-binned images the author additionally found that some algorithms cannot be classified at all, due to the fact that they provide error values for both accuracy classes. **Table 4** shows the mean error of each focusing algorithm compared with a manual measure as well as the standard deviation within a corresponding class. The error distributions of the algorithms were evaluated using Q-Q plots. The result was that the algorithms which could be classified uniquely were normally distributed, whereas the others algorithms which were unclassifiable clearly are not distributed normally. The unclassifiable algorithms therefore were excluded from further evaluations. As shown in **Table 2** and **Table 3** only the 12 remaining algorithms were evaluated. This leads to the conclusion that algorithms which are sensitive for a different quantity cannot be compared. Each carrier in the histogram shown **Figure 66** in represents an

accuracy class of characteristically different algorithms. Which means each offset of a carrier can be explained by the fact that the sampled characteristic is maximized in a certain plane in a dependable way. This characteristic is then the main characteristic of the algorithm class (i.e. for contrast functions, the frequencies; for power function, the intensity of the image).

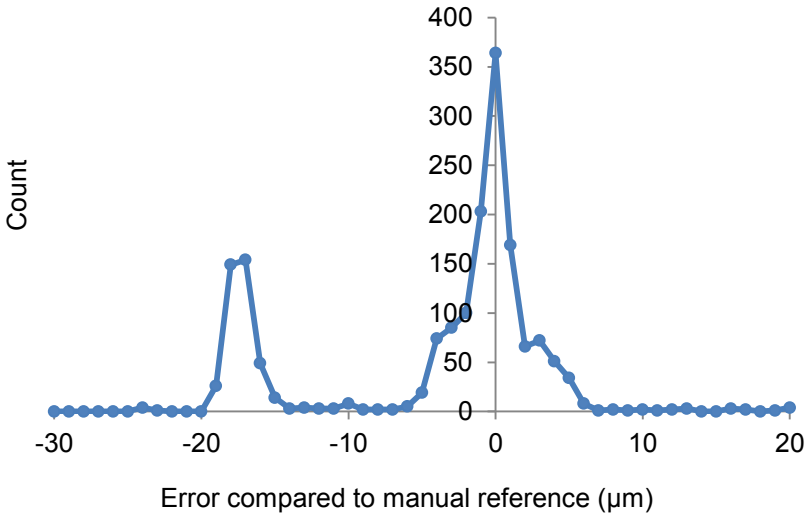


Figure 66 Error histogram of focusing algorithms

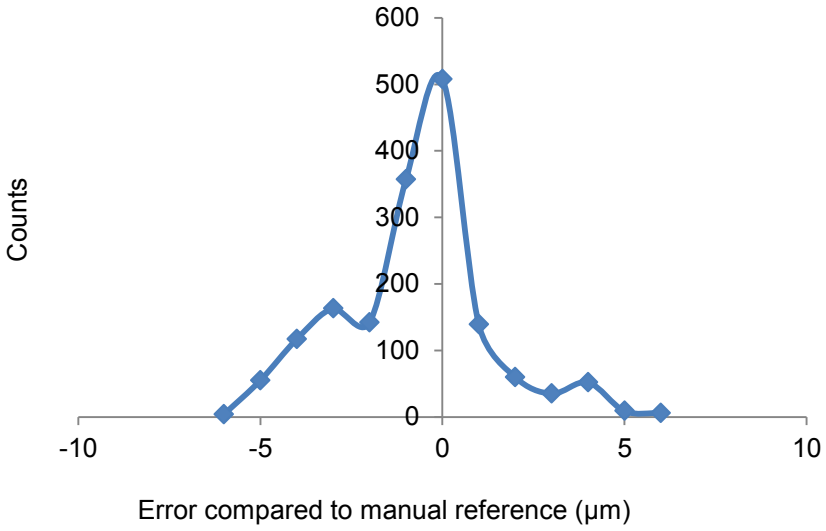


Figure 67 Histogram of all errors of all focusing algorithms for binned images.

Table 4 Accuracy classes of focusing algorithms

Alg.class	**Start**	**End**	**Focusing algorithm**	**Count**	**Mean**	**StdDev**
class 01	-32	-7	*Energy_laplace*	*45*	*-17,67*	*2,21*
			Abs_gradient	91	-17,65	1,76
			Laplacian	98	-17,60	1,72
			Laplacian_variance	*84*	*-17,56*	*1,70*
			Hist_entropy	*81*	*-15,98*	*2,95*
class 02	-7	20	Abs_variance	99	-3,46	1,40
			Energy_laplace	55	-1,04	2,81
			Image_power	91	-0,82	2,51
			Vollath_f5	91	-0,82	2,51
			Variance	99	-0,63	2,55
			Edgegradient	98	-0,41	2,14
			Genetic programming	99	0,29	2,68
			Squared_gradient	99	0,97	1,83
			Laplacian_variance	16	1,00	3,27
			Tenengrad	99	1,04	2,65
			Netten_filter	99	1,06	1,87
			Vollath_f4	99	1,18	2,62
			Brenner_gradient	99	1,27	2,77
			Sobel_tenegrad variance	99	1,29	2,72
			Hist_entropy	19	5,26	7,16

Table 5 Algorithm classes with binned data.

Alg.class	**Start**	**End**	**Focusing algorithm**	**Counts**	**Mean**	**Std.dev**
class 01	-6	-2	Tenengrad	100	0.20	1.48
			Vollath f4	100	0.38	1.45
			Brenner Gradient	100	0.39	1.50
			Netten Filter	100	0.49	1.51
			Sobel Tenegrad Variance	100	0.52	1.62
			Squared Gradient	100	0.54	1.49
			Laplacian	100	0.54	1.31
			Edge Gradient	100	-0.61	0.68
			Hist Entropy	100	0.61	9.39
			Laplacian Variance	100	-0.73	3.11
			Energy Laplace	100	-0.79	2.42
			Abs Gradient	100	-0.80	0.77
class 02	-2	6	*Genetic Programming*	*100*	*-1.43*	*1.73*
			Variance	*100*	*-2.32*	*1.29*
			Image Power	*100*	*-2.91*	*2.05*
			Vollath f5	*100*	*-2.93*	*2.04*
			Absolute Variance	*100*	*-3.96*	*0.82*

Outlier detection using the surface model as a reference

Due to dust, noise, and other unspecific signals the local focus plane finding can be irritated. Even the top five focusing algorithms evaluated showed outliers (**Table 6**). For normally distributed functions 99.73 % of all measurements will lie within $\pm 3\sigma$ of the Gaussian distribution. Outliers can be defined therefore as values outside the $\pm 3\sigma$ range. The detection of outliers is important due to different aspects. The automatic selection, as well as the classification of focusing algorithms, need to have dependable statistics. Above all the data analysis needs to be able to automatically exclude outliers, to provide dependable results. Therefore, in order to detect outliers, a model of the low frequency surface can be used. Presuming that the objects of interest are distributed around a low frequency waviness of the surface, the surface reconstruction explained above can be sufficient to estimate the amplitude of the error of the high frequency distribution around the focus plane. For the evaluation a fixed threshold of 5 µm (on average $\geq \pm 3\sigma$) was used to make the results more comparable.

Accelerating the focusing step using sub-sampling

In order to speed up the focusing step previous knowledge gained by incrementally modeling the surface helps to scan the specimen from an optimized starting position at each sampling point. Furthermore the focusing can be accelerated by terminating the focusing as soon as possible after detecting the global maximum of the focusing algorithm. The idea is to model the surface of the slide with a small but sufficient number of sampling points to start the focusing step of the interpolated sampling points with a height map. After the in-focus position was measured, the global maximum search should determine the in-focus plane as fast as possible. To decrease focusing time, the maximum search must be able to detect the global maximum “on-the-fly”, which needs a dynamic extremum search [143] and a model-based curve fitting [144], without knowing explicitly which values might appear outside the scanned volume. A maximum is generally characterized by the fact that the predecessor and successor values are smaller than the current one. Since a microscope normally has to work for a

variety of samples, the system design of an autofocus system has to be stable against multimodal functions however. Local and global maxima hence have to be discriminated. The presumption used here is that the global maximum has a higher steepness than a local maximum. A global maximum peak can contain more than one value. Measuring the steepness between the current value, two values before and two values after the current value helps to dependably find the maximum. Ambiguity in this case often comes from dust particles which can generate false global maxima.

Figure 65 illustrates the workflow how to accelerate the focusing step. First one well is sampled with an equidistant grid of 10x10. Second the best focusing algorithm has to be determined automatically. Because the automatic selection is based on the ranking of the standard deviations of the focusing algorithms, this step ensures two fundamental principles. The first principle is to maximize the number of in-focus images, and the second is to minimize the standard deviation, which will influence the actual acceleration of the autofocus system. To find the best focusing algorithm therefore is a fundamental step towards solving the conflicting goals of speed and quality. The third step has to determine the fastest sub sampling rate. The best focusing algorithm, as well as the sub sampling rate, will finally be taken for all others wells. Optimizing the focus starting position to be as close as possible to the in-focus plane helps to avoid wrong maxima detections, because dust often can be found on the surface of liquid rather than on the surface of the carrier. The acceleration is calculated based on the following formulas:

$$measurements_{static\ case} = n\frac{\mathrm{z}_{range}}{stepwith} \tag{Eq-20}$$

$$measurements_{dynamic\ case} = n_1\frac{\mathrm{z}_{range}}{stepwith} + (n-)\frac{3 * \sigma}{stepwith} + 2 \tag{Eq-21}$$

$$acceleration = \frac{\text{number of adaptive measurements}}{\text{number of static measurements}} - 1 \tag{Eq-22}$$

Table 6 Outlier detection statistics using the surface model and a fixed threshold of 5 µm.

	Number of outliers				
Focusing algorithm	Well01	Well02	Well03	Well04	All wells
Edge Gradient	3	2	0	0	1.25
Absolute Gradient	15	1	0	0	4.00
Laplacian	3	3	0	0	1.50
Laplacian Variance	30	7	5	8	12.50
Squared Gradient	2	4	2	0	2.00
Vollath F4	4	3	3	1	2.75
Netten Filter	2	2	2	1	1.75
Brenner Gradient	4	1	2	2	2.25
Tenengrad	4	1	1	1	1.75
Energy Laplace	n.a.	8	2	4	4.67
Sobel Tenegrad Variance	3	3	1	0	1.75
Genetic Programming	4	3	1	1	2.25

Table 7. Accelerations factors based on standard deviations. The average is built out of well 02-04 only because Well01 shows bad results because the Absolute Gradient is not suitable for non-binned images.

	Standard deviations			Acceleration			
Nr. of sampled spots	Well02	Well03	Well04	Well02	Well03	Well04	Average
4	3.67	3.14	3.23	440%	329%	298%	355%
9	3.22	1.94	2.08	375%	392%	356%	**374%**
16	2.72	0.84	0.98	294%	387%	362%	347%
25	2.73	0.77	0.86	199%	248%	240%	229%
36	2.73	0.74	0.85	131%	156%	151%	146%
49	2.71	0.75	0.86	83%	94%	92%	90%
64	2.68	0.73	0.84	47%	52%	51%	50%
81	2.65	0.73	0.84	20%	22%	22%	21%
100	2.65	0.72	0.83	0%	0%	0%	0%

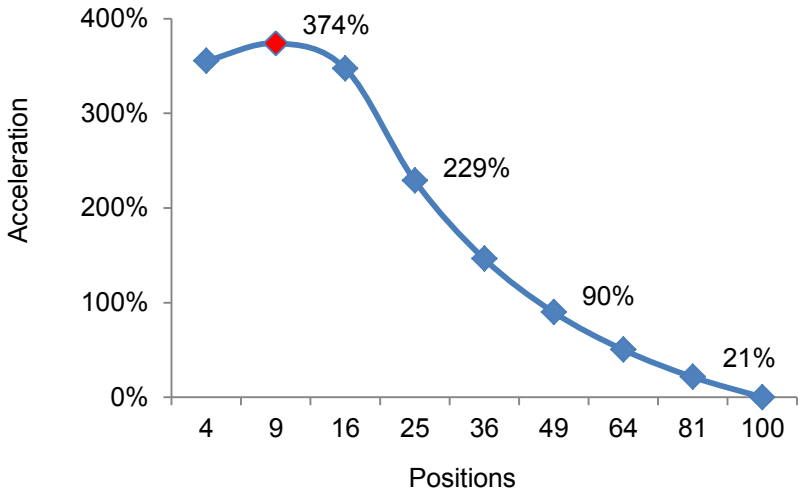


Figure 68 Acceleration of focusing by approximating the surface. The less positions are measured, the higher the approximation error is, but the higher the overall acceleration becomes. An acceleration of 374% could be reached if only 9 positions are measured and 91 positions focused based on an estimated focus position.

4.5.4 Results

Surface modeling

The first and second order approximation caused systematic error; whereas a third order polynomial was sufficient to minimize the average error of the approximation under the optical depth of field of 1 µm and within the range of the manual measure standard deviation (1 µm). In the implementation an incremental approach was used to asynchronously recalculate the surface model with each new measured focus position, which helped to gradually improve the precision of the approximation and make the estimated focus positions more dependable.

Automatic selection of the best focusing algorithm

The number of in-focus images within the top 12 algorithms could be doubled from 45% (genetic programming) to 90% (edge gradient), although the manual evaluation showed an average accuracy of the focusing algorithms which was almost the same (**Table 1**). The new focusing algorithm, which the author called Edge Gradient, was ranked among the best algorithms. It could be shown that it is stable against noise in non-binned images, whereas the Absolute Gradient failed for noisy images. With the Edge gradient algorithm, the highest amount of in-focus images was obtained. However using the automatic approach the Absolute Gradient performed best for binned-images. In order to automatically select the best focusing algorithm, a pre-classification of the algorithms is necessary. It was shown that only 12 out of the 17 algorithms evaluated belong to the same analytical class as the manual measure by the human eye. In addition the surface model was a suitable reference to detect and exclude outliers after selecting and using the best focusing algorithm.

The main result of the comparison shown in **Table 2** is that the automatic approach using a surface model is positively correlated to the manual approach, which proves that it can be used to automatically select the best focusing algorithm. The ranking of the automatic approach is slightly shuffled however. It

can be assumed that the manual measure has errors itself; the ranking after the amount of in-focus images is tendentially correct related to the manual measure as the ground truth. The precision of the manual measure was below the depth of field of 1 µm. It can be assumed therefore that the chosen algorithms catch the correct focus plane within the ambiguous range of 1 µm. The best focusing algorithm by the number of manually counted in-focus images was the Edge Gradient, which was introduced by the author, with 91.67% in-focus images. The second best focusing algorithm in this ranking was the Absolute Gradient, with 89.33% images in-focus. These values are very close to each other, however subtracting the error of the manual measure as a reference could alter this close ranking in a way that the ranking will become identical to the automatic one. The automatic ranking using the surface modeling as a reference showed that the Absolute Gradient is the best algorithm for binned images. Furthermore, the evaluation showed that ranking after the accuracy is not suitable, because it is not correlated to the number of in-focus images. The precision of focusing algorithms in contrast gives a dependable impression as to how many images will be in-focus, once the accuracy of the algorithms is below the depth of view. The problem of the precision measurements on the other hand is that outliers can influence the mean value, and thus the standard deviation. Outliers therefore need to be excluded automatically in order to receive dependable statistical variables.

Acceleration

The maximal acceleration which could be reached was 374% with a second order fitting of the surface as can be seen in **Figure 68**. The detailed precision measurements are described in **Table 7**.

4.5.5 Conclusion and Discussion

This chapter presents an approach for how to automate the selection of the best focusing algorithm for experiment specific data, and in combination with outlier detection improve the number of in-focus images. The surface modeling also was evaluated to provide a basis for the acceleration of the focusing step. It was shown that focusing algorithms evaluated can differ in the number of in-focus images by a factor of two. After automatically choosing the best suitable algorithms, the focusing step could be accelerated three- to fourfold.

Our original hypothesis was that a surface modeling can dump our precision errors and reconstruct the focal plane. The author has compared his method with a manual evaluation, and it could be shown that the automatic selection of the most precise algorithm is possible, and our hypothesis holds true. To be able to do this, however, the algorithms have to be within the algorithmic class of the manual measure, meaning that their accuracy is below the depth of view. The author showed that some evaluated algorithms display systematic errors, from which he derived that they measure different qualities that our eye does. Finally, reducing the number of images taken in the focusing step decreases phototoxicity and bleaching, and increases the throughput, especially for time lapse experiments. Our method addresses this critical part of microscopy and can successfully reduce the amount of necessary images to find the focal plane, while maintaining the maximum amount of in-focus images. Our method could be widely implemented, not just based on nuclei, but also on all applications where the surface bounding has a low frequency and the biological offset of the marked part of the specimen is constant. In addition, in the future this knowledge could be shared in a cloud, in which algorithms are available as image quality improvement strategies for different kinds of experiments. Operators would then not just benefit from a local selection of image quality improvement strategies, but rather from prior and shared common knowledge.

4.5.6 Appendix

A1 Focusing algorithms

Absolute Gradient [**147**]. This algorithm sums up the absolute value of the first derivative:

$$F_{abs\ grad} = \sum_{Height} \sum_{Width} |i(x+1,y) - i(x,y)| \qquad \text{(Eq-23)}$$

Squared Gradient [**147**]. This algorithm sums squared differences, making larger gradients exert more influence:

$$F_{sq\_grad} = \sum_{Height} \sum_{Width} (i(x+1,y) - i(x,y))^2 \qquad \text{(Eq-24)}$$

Brenner Gradient [**46**]. This algorithm computes the first difference between a pixel and its neighbor with a horizontal/vertical distance of 2.

$$F_{Brenner} = \sum_{Height} \sum_{Width} (i(x+2,y) - i(x,y))^2 \qquad \text{(Eq-25)}$$

Tenenbaum Gradient (Tenengrad) [**46, 148**]. This algorithm convolves an image with Sobel operators and then sums the square of the gradient vector components.

$$F_{Tenengrad} = \sum_{Height} \sum_{Width} S_x(x,y)^2 + S_y(x,y)^2 \qquad \text{(Eq-26)}$$

Laplacian [139]. This algorithm convolves an image with a Laplacian convolution mask of:

$$L = \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix} \quad \text{(Eq-27)}$$

Laplacian Variance [149]. "Continuing with these approaches, the best-in-focus image f_{BF} in the stack, according to the Laplacian gradient magnitude variance, will be the image with highest variance, in this context

$$f_{BF=}\{f_k \; where \; MAX(N(LPVFM_k)) \quad \text{(Eq-28)}$$

where LPVFM$_k$ is a vector containing normalized values." **[149]**

Energy Laplace [150]. This algorithm convolves an image with a Laplacian convolution mask:

$$L = \begin{bmatrix} -1 & -4 & -1 \\ -4 & -20 & -4 \\ -1 & -4 & -1 \end{bmatrix} \quad \text{(Eq-29)}$$

Vollath's F4 [151]. This algorithm is defined as:

$$F_{VOL4} = \sum_{Height} \sum_{Width} g(i,j)\, g(i+1,j) - \sum_{Height} \sum_{Width} g(i,j)\, g(i+2,j) \quad \text{(Eq-30)}$$

where g(i, j) is the gray level of. This algorithm computes the autocorrelation and is robust against noise.

Sobel-Tennengrad Variance [149]. **"**The best-in-focus image f_{BF} in the stack based on the Sobel-Tenengrad gradient magnitude variance will be the image with highest variance in the sense

$$f_{BF=}\{f_k \; where \; MAX(N(STVGM_k)) \quad \text{(Eq-31)}$$

where STVFM$_k$ is a vector containing normalized values" [149]

Netten filter [152]. The filter is computed with

$$F_{Netten} = \sum_{Height} \sum_{Width} (i(x+1,y) - i(x-1,y))^2 \quad \text{(Eq-32)}$$

Genetic Programming [153].

Edge gradient. This thresholded algorithm sums the gradients at the edges of the foreground:

$$F_{Edge} = \sum_{Height} \sum_{Width} (foreground(x_1,y_1) - background(x_2,y_2))^2 \quad \text{(Eq-33)}$$

where foreground $>$ (mean+ $x\sigma$) (here x is 0.5) and background $<$ mean. x_1, y_1 are pixels which were classified as background. x_2, y_2 are the nearest neighbor pixels of x_1, y_1, which are classified as foreground.

4.6 RFT (Re-Focusing Trigger)

4.6.1 Abstract

Autofocusing is an elementary technology for automated imaging. Crucially, the significance of the analysis output is a function of the initially acquired raw data and therefore strictly controlled in order to minimize error propagation. In high-throughput microscopy the balance between image quality and acquisition rate is a fundamental optimization step. The auto focusing step represents a relevant target for increasing the throughput of microscopes and the sampling rate for live cell screens. Improved focusing methods have the potential to both minimize photobleaching and phototoxicity, and increase sampling speed. Avoiding unnecessary focusing steps therefore is a central challenge to optimize the process. In live cell imaging the focusing step has to be performed on run time in order to ensure that images are still in focus. This lowers the throughput of the microscope and increases phototoxicity. We propose a refocusing trigger, which triggers a refocusing only on demand. This logic was implemented in the LifeXplorer framework, offering highly dependable logic execution. It was tested with an Olympus IX 81 as well as with a Nikon Ti-E fluorescence high-throughput microscope. The refocusing speeds up the imaging throughput up to threefold and could reduce the phototoxicity by half. This method can easily be integrated to existing microscopy environments.

4.6.2 Introduction

Rapid, high-resolution imaging is emerging as an essential tool in quantitative biology. Since data analysis depends on the incoming data, the data quality of the raw images in light microscopy is highly relevant for effectively gaining insights into biology. In high-throughput microscopy a conflicting goal is to provide fast imaging on one hand, and on the other hand to retain the quality of the data acquisition. One of the most important perquisites for high-quality data is precise focusing. The accuracy of focusing algorithms differs with different data and therefore the choice of the best algorithm depends on the experiment. This can be seen in the literature [2, 3, 139] where the evaluation of focusing algorithms has shown different results, depending on varying data. Within one image, optimized estimations of the focal planes of single sub areas can be computed. Image stacks can be cut into sub areas, and for each of these areas the focus position can be calculated separately [154]. This has in addition shown that the focus position is related to the physical unevenness. In order to speed up the focusing step, a sub sampling of the observed area can be used [141]. Several patterns, i.e. a regular grid or more sophisticated ones using Halton sampling points, can be used to increase speed. In addition different adaptive algorithms with variable z-steps can be used to accelerate the focusing step [1]. Pre-processing the acquired raw data before computing the in-focus plane was analyzed to be suitable to generalize the usage of focusing algorithms [142]. All published methods however focus on optimizing the focusing algorithm, but do not answer the question if, and especially to which extent, the re-focusing in live cell imaging step itself is necessary. From the ARCO principle it can be derived that also the scheduling of a task itself can be simulated first, in order to check if the result-costs ratio is higher if a task is performed or not.

A refocusing trigger therefore is proposed in this thesis, which triggers a refocusing only on demand. It is based upon the ARCO principle. The refocusing trigger logic was implemented in the LifeXplorer framework. It was tested with an Olympus IX 81 as well as with a Nikon Ti-E fluorescence high-throughput microscope. The refocusing speeds up the imaging throughput up to threefold and

could reduce the phototoxicity up to twofold. The method can easily be integrated into existing microscopy environments. In order to provide high available logic execution, specialized bindings were introduced into the LifeXplorer framework. The controller logic of the organic computing network was modified such that memory afflicted workflows can be run redundantly, synchronizing necessary information between each other or alternately computing a workflow completely redundant on runtime. The last method, which can be called a hot standby, ensures that the refocusing trigger is computed in parallel and the task controller in the microscope can identify redundant signaling.

4.6.3 Material

Hardware, Software and Development Environment

The experimental setup consists of a motorized inverted microscope (Olympus IX81), a frame grabber (Matrox Meteor-II) and a CCD camera (Hamamatsu C9100-02). The motorized inverted microscope has a minimum focus step width of 0.01 µm. Two objectives were used, a 10x objective (Olympus UPlanFL, 3 µm depth of field, NA 0.3) and a 40x objective (Olympus LCPlanF1, 1 µm depth of field, NA 0.6). The microscope was observing in the brightfield mode. The focusing workflow employs two different axial steps for different magnifications [1]. 200 sampling points were measured with 5 µm axial steps using a 10x objective and 1 µm axial steps using a 40x objective. The 40x images were taken without binning and 60ms exposure time. For lateral sampling a 10x10 regular grid with an accuracy of 900 µm was taken in each chamber. Two wells were imaged. The sampling time was 15 min. The automation of the image acquisition routine was implemented in C# and C++ using the native serial port commands of the specific hardware units. Both were integrated using our distributed microscopy framework called LifeXplorer.

Biological samples

Human MCF7 breast cancer cell lines (Cell Line Services, Heidelberg) were maintained in DMEM/10% FBS/L-glutamine/non-essential amino acids / penicillin / streptomycin / amphotericin B (Invitrogen). Cells were plated in 8-well microscopy µ-slides (iBidi) at a density of 20,000 cells per well. To label nuclei, cells were incubated for 15 min with Hoechst 33342 at a concentration of 1 ug/mL at 37C.

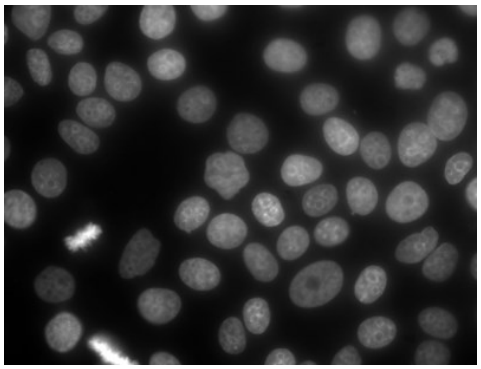


Figure 69 Cell nuclei of MCF7 cells

4.6.4 Methods

The refocusing trigger logic is based on the idea that focusing is only necessary if the quality of an image drops under a configured threshold. The quality can be measured with diverse algorithms published in the related literature [2, 139]. In this thesis the Absolute Gradient and the Brenner Gradient were taken.

Absolute Gradient [117]. This algorithm sums the absolute value of the first derivative:

$$F_{\text{abs grad}} = \sum_{\text{Height}} \sum_{\text{Width}} |i(x+1,y) - i(x,y)| \qquad \text{(Eq-34)}$$

Brenner Gradient [46]. This algorithm computes the first difference between a pixel and its neighbor with a horizontal/vertical distance of 2.

$$F_{Brenner} = \sum_{Height} \sum_{Width} \left(i(x+2,y) - i(x,y)\right)^2 \qquad \text{(Eq-35)}$$

As it can be seen in **Figure 70**, after the first focusing step, the value of the quality of the in-focus image is stored. In the next runs, the focusing step is not automatically performed anymore. Instead images on each configured sampling point are taken and the quality of the image is always measured in addition. The value of the image quality is then related to the initially measured value. If the currently measured quality drops below a configured threshold, i.e. 90% of the initial image quality, the refocusing trigger logic triggers the microscope to refocus a position. Two different behaviors can be configured regarding how to treat the existing history after the trigger signal is sent to the microscope.

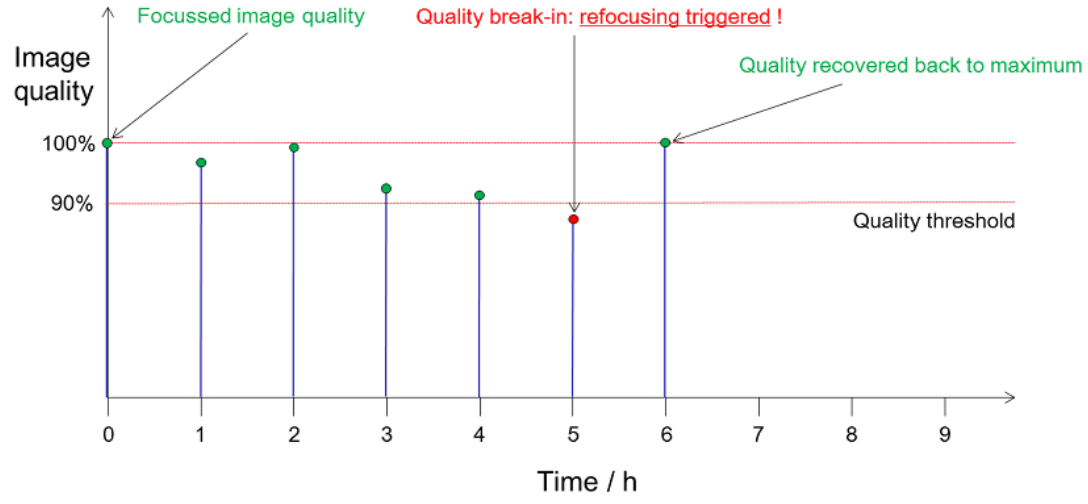


Figure 70 Contrast quality over time.

History management

Once the logic detects that a refocusing is necessary:

1. The history of the trigger logic is deleted so that the new value measured after refocusing the position is considered as being the initial measurement again.

2. The initial value stays the same

The first behavior is supposed to be the default behavior, since it is not clear how the fluorescence signals change over time, or if the initially measured value is also suitable as a constant reference over the time. In principle, it is only important that the reference value comes from an image in focus. The second behavior however is possible if the quality of the image over time is controlled by additional methods. I.e. if ASEC (Application Specific Exposure Control) is activated and configured to stabilize the Signal/Noise ratio over time, the initial value can be kept, since ASEC will try to force this value to remain stable over time. In this case it becomes obvious how different methods have to be harmonized and the execution of tasks has to put into a well-defined order. ASEC i.e. has to be run after the focusing logic is completed. Otherwise the re-focusing trigger logic might not detect a focus drift, because the overall contrast of an image is stabilized by the re-adjustment of the light exposure.

Trigger scheduling

As it can be seen in **Figure 73**, received trigger signals are stored in a buffer first. Depending on the configuration, the scheduler can then decide if refocusing a position is directly performed after the current position is imaged, or alternately if all trigger signals in the buffer are handled after the current imaging run is finished.

Once a trigger is received, two different behaviors can be configured:

1. Schedule refocusing of the position as the next task
2. Schedule refocusing of all positions in the refocus trigger buffer after all positions are scanned.

Both scheduling alternatives can cause completely different behaviors. The first configuration is more powerful, because it always allows all possible configurations of the refocusing management as follows. After a position is refocused, a measured offset between the old and the new position can be added to other positions, without explicitly refocusing them. Depending on the real-time conditions, the first behavior however can be critical, since the sampling time of

the positions that are imaged afterwards can be changed. For the second behavior, in the spare time after the imaging is done, the refocusing triggers can be performed. The advantage of this behavior is that only the positions which need to be refocused will have a dynamic sampling time. The disadvantage in contrast to the first behavior can be that one single refocusing step causes that all positions which are taken afterwards will not be in real-time anymore. For both behaviors the sampling rate should be higher than the actual imaging time. How much higher is obviously an estimation based on how much refocusing is necessary, and certainly needs to be based on previous knowledge about the experiment.

Refocusing management

Once a position is refocused, three different behaviors can be configured:

1. The focus is changed only for the refocused position
2. The focus is changed for all positions of the same well
3. The focus is changed for all positions

The first behavior is the default behavior. Each position is treated individually and the triggers do not correspond to each other. The handling of this behavior is the least complex one. In the case of redundant trigger logic execution, trigger signals for each position in the same run have to be processed only once to avoid unnecessary refocusing on positions which were refocused already. The second behavior however causes a more complex handling of the trigger signals. The task scheduling in addition has to be aware of this behavior as mentioned above. Once positions are changed either by a physical or a computational refocusing, triggers for this position should be discarded. In the scheduler logic each signal has a unique key and meta information, including the task, the position, the well, the channel, the configuration and the run index. Having configured the propagation of the measured focus offset to other positions, tasks can virtually be created in the scheduler as if these positions had actually been refocused. As the scheduler

by default will not perform the exact same task twice, in order to be stable against redundant requests, it will then automatically discard all incoming triggers for positions which are changed computationally.

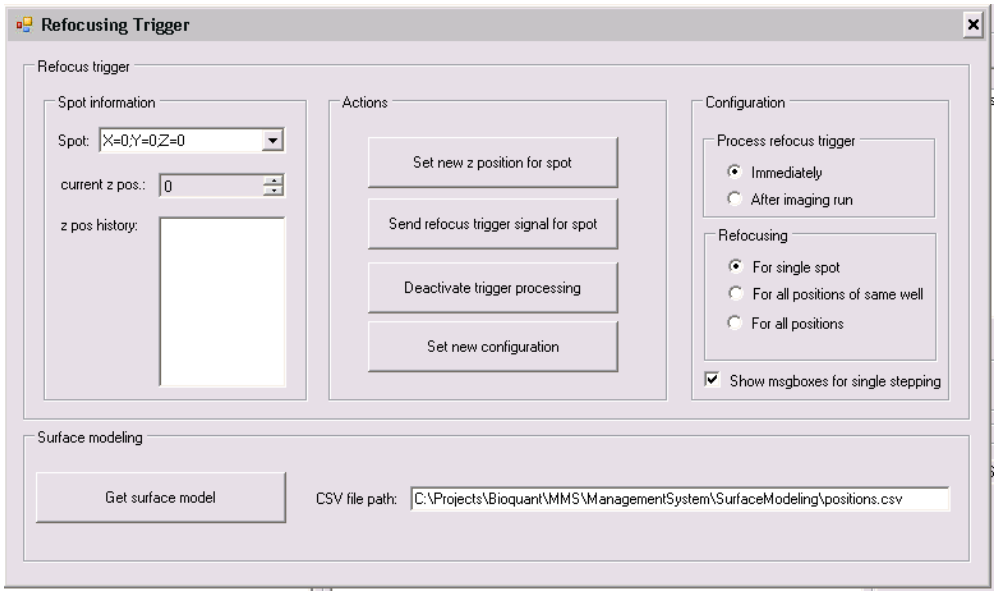


Figure 71 Refocusing trigger configuration form

The behaviors explained above can be configured in the Life Explorer Microscope Management System by an administrator. The form shown in **Figure 71** in addition makes it possible to simulate trigger signals, execute them manually, and track the Z positions over time. In principle it was integrated to run the experiments mentioned in the results part to get a first impression about the possible options.

Redundancy management

Figure 72 illustrates how the redundancy management for the refocusing trigger logic is implemented. Each sampling point is routed to its own subnet. This practically means that the whole computing workflow for all optical channels is always handled by the same nodes for the same sampling point. By default, the controller of the computing network, in this case the LifeXplorer

computing cloud controller will dynamically push tasks to resources with the lowest workload. For memory afflicted workflows however the dynamic scheduling is not suitable. Once a node stores historical information used by the workflow logic (here the refocusing trigger) the scheduler has to bind the sampling point to this node. At the same time redundancy for highly available workflow execution should be possible. As can be seen in **Figure 72** the green arrows lead to the same node. In the LifeXplorer framework it is possible to fix the binding between a module and a node. The last module in the case of the refocusing trigger logic is the decision support system, which finally sends a trigger signal to the microscope, or not. This module depends on historical information about the image quality as explained above. A module in addition can be marked as highly available in the graphical user interface. It will then get a fixed list of computing nodes, on which it will be executed. Practically this means that the controller will not always schedule a processing task redundantly on the exact same nodes. In addition, the routing information for the preceding processing steps needs to be adapted so that the relevant inputs are linked redundantly to the corresponding nodes. **Figure 73** visualizes the communication topology in more detail with all the relevant modes within the microscope control logic, along with the processing logic for the refocusing trigger.

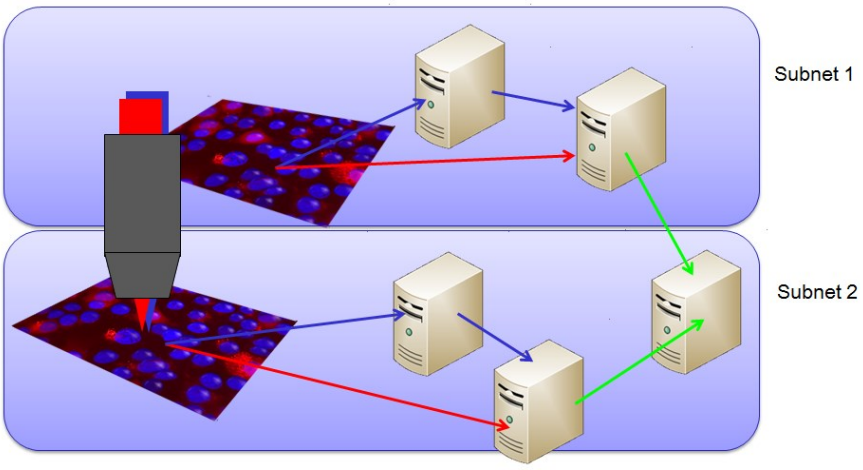


Figure 72 Different sampling points are computed in seperate subnets

On the microscope host computer the hardware control logic is executed. The basic imaging workflow is to focus a position, to scan it, and attach the image with a configured task to a distribution worker. This worker sends images and receives tasks. This unit is completely asynchronous to the imaging workflow. After the image is scanned, the next spot can be scanned or the workflow pauses. After it has paused, the scanning is continued if the refocus buffer of the task manager is empty. If it is not empty, the refocusing step is performed. If and when exactly the refocusing is done depends on the concrete system behavior, explained above. The distribution worker sends the image of a sampling point to the virtual microscope unit, which normally is run either on the microscope host node as well or as closely as possible, in order to avoid unnecessary communications delays. The virtual microscope unit integrates three essential sub modules, a receiver, a router and a feedback manager. Receiving the image from the sender unit of the distribution worker on the microscope, the controller logic will have to create binding between the sampling point and at least two different nodes in order to ensure high availability of the refocusing trigger logic. Measuring point 1 in **Figure 72** will be bound to worker2 and measuring point 2 to worker 3. In addition the redundant "hot spot" for the trigger logic of measuring point 1 will be worker 3. This can be seen in the figure by the fact that the calculated gradient in worker 2 will be sent to the history manager of both worker 2 internally and worker 3 in addition. This means that the last image quality stored is always saved on two different nodes. If the node of worker 2 fails, the observer of the LifeXplorer network will detect it and communicate it to the controller. The controller then will switch the task processing for measuring point 1 to worker 3.

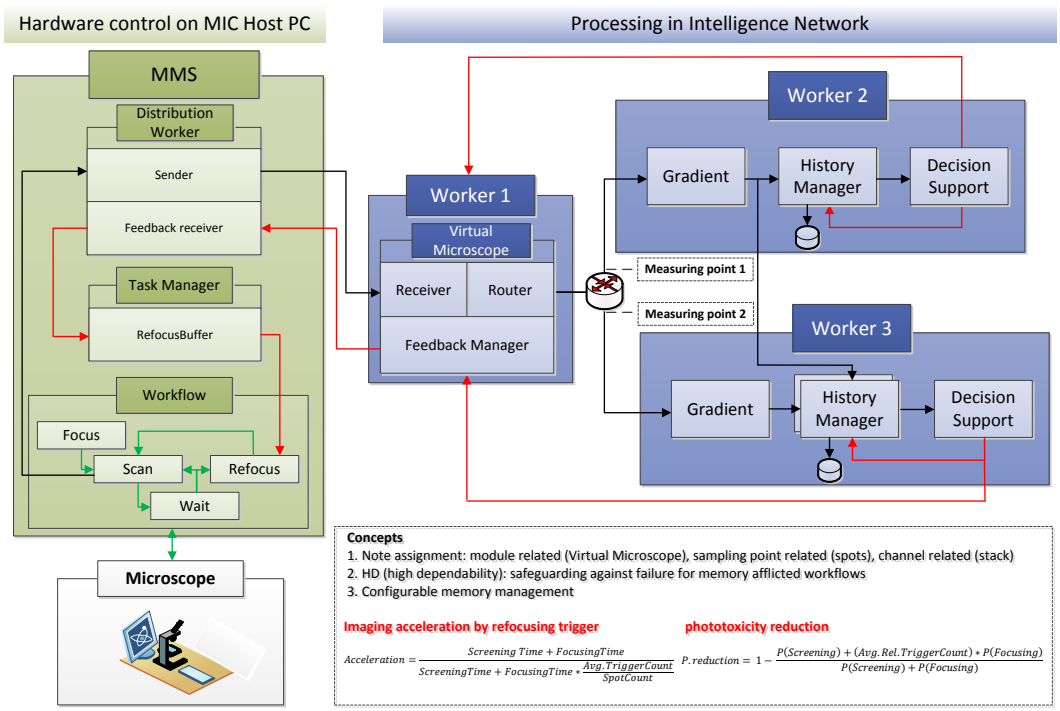


Figure 73 Refocusing trigger implementation in all LifeXplorer components

Since the history manager of worker 3 stores the relevant information for measuring point 1 already, no loss of information is caused by the breakdown of worker 2. The decision support system of worker 3 will then be executed twice with two different inputs for both measuring points. The refocusing trigger signals will be sent to the virtual microscope worker, where they can be passed through or sorted out in order to avoid double tasks in the microscope control logic. Both are possible, since the microscope task scheduler will additionally sort out double entries. To evaluate the refocusing trigger logic, two basic measurements were taken. The time to return and the frequency of the triggers were measured. Both evaluation results are presented in the next chapter. In the materials part, the hard- and software setup as well as the biological sample are described.

4.6.5 Results

In **Figure 74** 200 sampling points were measured with a sampling rate of 15 min over 6 hours. The refocusing trigger logic was configured to fire an event once the image quality drops below a relative value of 95%. The quality of an image was measured with the Brenner Gradient [46]. Once a trigger signal was sent, the current run index minus the last run index of the last trigger signal was saved. Later these frequencies were multiplied by 15 again in order to scale them with the sampling rate. The histogram of all frequencies was plotted then in **Figure 74**. As can be seen here, the distribution of the trigger frequencies can roughly be approximated with a Gaussian distribution curve. The average frequency was 172.3 min. 251 triggers were sent over time, which is equal to 5% of the 4800 images taken. This means that the refocusing was only necessary on average every ~180 min and only in 5% of the cases.

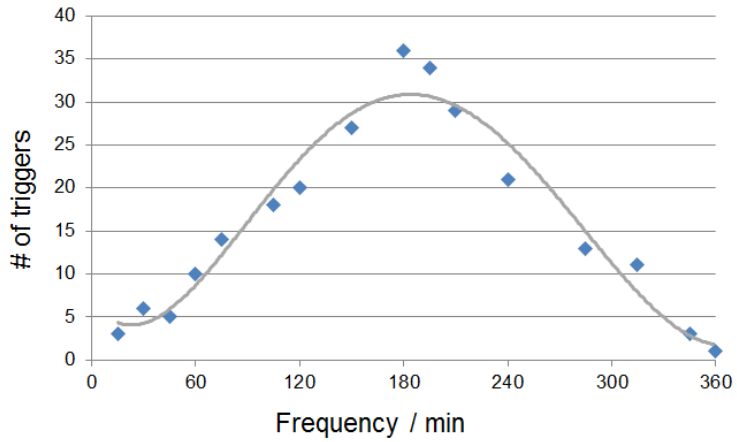


Figure 74 Trigger distribution over time.

The acceleration of the imaging can be approximated with:

$$Acceleration = \frac{Screening\ Time + FocusingTime}{ScreeningTime + FocusingTime * \frac{Avg.TriggerCount}{SpotCount}} \qquad \text{(Eq-36)}$$

The measured results lead to the following acceleration:

$$Acceleration = \frac{5min + 10min}{5min + 10min * \frac{10}{200}} = 2.73 \quad \text{(Eq-37)}$$

The acceleration by the refocusing trigger was 2.7 fold.

The phototoxicity reduction can be approximated with:

$$P.reduction = 1 - \frac{P(Screening) + (Avg.Rel.TriggerCount) * P(Focusing)}{P(Screening) + P(Focusing)} \quad \text{(Eq-38)}$$

P $phototoxicity$

The measured results lead to the following phototoxicity reduction:

$$P.reduction = 1 - \frac{1 + 0.05 * 2.5}{1 + 2.5} = 71\% \quad \text{(Eq-39)}$$

On average the refocusing algorithm takes ten images, binned. The refocusing trigger logic works on the primary imaging data, which were non binned images. The exposure time of the focusing step is then reduced fourfold in relation to the imaging exposure time, since the focusing is done on 2x binned images. In the same time ten instead of only one image is taken, which leads to a total exposure time of the focusing that is 2.5 higher than the imaging to take one single plane, as it was done in the experiment. It was furthermore assumed that the phototoxicity is on average linearly related to the exposure time, which means that the reduction of the exposure is directly related to the reduction of the phototoxicity. Taking these assumptions and the numbers mentioned above, the phototoxicity is reduced 71%.

To calculate the latency of a refocusing trigger signal, the Brenner Gradient [46] was taken. The Absolute Gradient [117] was not considered anymore after it become obvious that it is not as stable against noise as the Brenner Gradient. The latency statistics are illustrated in **Figure 75**. The measured deviation of

the latencies can be approximated with a Gaussian distribution curve. The average latency was 707 ms. The standard deviation is ~200 ms. This deviation is caused by the fact that software is run on a normal Windows operating system where others tasks are executed in parallel and no real-time ability is guaranteed by the scheduler. However the average latency is still fast enough to suit a soft real time of 1 second on average. This means that the trigger logic on average can react below 1 second. The focusing on average took twice the time of the imaging time. Having on average only 5% of the spots which need to be refocused, the spare time for the imaging run in addition only needs to be 10% of the sampling rate. Although the refocusing trigger logic was hosted only on one single node, the performance was still suitable enough to cover soft real-time needs.

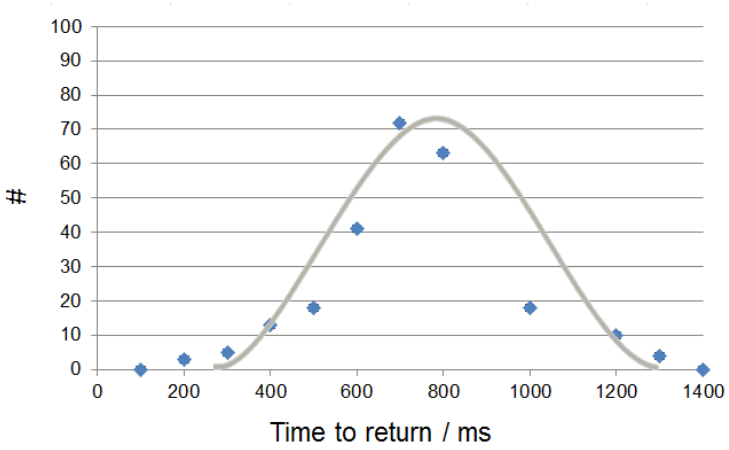


Figure 75 Trigger latency distribution

4.6.6 Conclusion

The refocusing trigger approach is a derivation of the ARCO principle. A refocusing step should only be executed if necessary. The result in this case is quality of the image. If this quality drops below a configured threshold, the costs to lose this quality can be configured as being high enough to rather send a trigger signal to the microscope in order to re-adjust the focus and thus the

quality of the image. The method was evaluated and it showed that the refocusing trigger could reduce the phototoxicity of the imaging 71% and at the same time accelerate the imaging throughput 2.7 fold. Only in 5% of the cases was a refocusing necessary. This means that depending on the application and the environment, the focusing step can be optimized due to the fact that it is only performed on demand. The approach of triggering a refocusing task is independent from the focusing algorithm itself. The possibility to reduce phototoxicity and to accelerate the focusing step in life cell imaging is very high. Besides algorithmic optimizations of the focusing itself, this optimization can help to increase the Result/Cost ratio immediately without changing the existing focusing algorithms. The relevant literature did not consider that the throughput of live cell imaging is significantly dependent on the refocusing step. It was furthermore not considered that the task to refocus a position is not always necessary, and that the need to refocus a position can be derived from the primary imaging data as well, without creating additional redundant data.

4.6.7 Discussion

The refocusing trigger introduced here is based on the idea of aggregating the quality of an image into one single number. Practically each object in an image can have its very own focus plane and the mean value for the whole image might not be suitable. This especially can be the case if selected objects are tracked over time, or if phenotypes of interest are tracked. Whole parts of an image therefore might not be interesting anymore to answer the actual scientific question. The focusing logic therefore has to be supported by region of interest algorithms and make decisions based on single objects, rather than on the whole image. Both the XY and the Z position might change over time and the focusing step has to be separated into two parts: a rough focus plane estimation and a single object based focus position estimation [154]. The rough estimation has to be performed in order to ensure that the region of interest (ROI) algorithms work as well as possible. After performing the ROI algorithms, the fine estimation of

single focus positions has to be executed. Integrating these steps into a microscope, it would be possible to do XY and Z tracking with the best quality possible. In addition, the microscope could make autonomous decisions about the number of Z planes on each position. If some objects drift out of the Z position where an image was taken, the imaging could automatically be expended and additional planes be taken. This way is would be possible to avoid taking unnecessary images, while ensuring that the objects of interest are always in focus in one of the acquired images. However the handling of the data would then need to be smarter. Either an artificial image would afterwards have to be created where all objects of interest are in focus, or the data analysis would need to have structured knowledge about objects and where to find them. Both can be critical since it increases complexity and thus the possibility of error propagation. In this case, scientists would not be easily able to evaluate the data by themselves. Even merging several focus planes into one image, most simply by a maximum projection [155], can cause several issues. For example, the image resolution is reduced, since the integrated volume is increased and all out of focus signals of blurred objects are integrated and overlapped as well. Artificially creating a merged image in any case needs a model that provides the lowest information loss possible and ensures that the data analysis will not detect artifacts of the merging. Eventually these investigations have to be done to reduce the complexity of life sciences for scientists by increasing the complexity of the decisions microscopes and others instruments autonomously make based on the ARCO principle.

Chapter 5
Conclusion and Future work

5.1 Conclusion

This study introduced the Automated Result / Cost Optimization (ARCO) algorithm according to standard experimental approaches, using existing microscopy platforms. Phototoxicity [5, 6] and photobleaching [7-9] create fundamental problems intrinsic to live-cell imaging and reduce image quality. It is widely recognized i.e. that light exposure results in mitochondrial dysfunction [33, 34]. Reduced illumination has been achieved via optimized pixel-dwell time with specialized laser scanning microscopy [10, 11, 35-37], and through the development of application-specific refocusing algorithms [1-3]. Integrative platforms, including Micropilot [4, 27], employ machine learning to extract and describe complex phenotypes. Machine-learning approaches can utilize generated data to analyze factors underlying data generation. Such approaches are emerging as critical tools in the field of quantitative biology [12, 28-32].

The effects of phototoxicity on mitochondrial bioenergetics and Golgi reformation demonstrated that *a)* ARCO optimized the imaging's quantitative result value twofold (object count and segmented area) (**Suppl. Figure 5**) and *b)* minimized the impact of the acquisition process on the observation. Phototoxicity was reduced 3-6 fold in both experiments. The evidence thus suggests that artifacts stemming from the imaging process were almost eliminated (**Figure 51** and **Figure 52**).

Implementing ARCO in LifeXplorer during run time enabled the experimenter to optimize dynamically the parameters. ARCO therefore renders obsolete costly trials of sequential data acquisition and evaluation. The decisions based on ARCO might appear as non-intuitive to the inexperienced user, as in the example demonstrating how images with a low signal–to-noise ratio result from the

optimization of illumination (**Figure 42**). This appearance stems from the fact that the algorithm optimizes quantitatively for the scientific question but not for bright images with a high dynamic range, as in the case of the user. Decisions to optimize the imaging workflow during runtime are based on an extensive knowledge of using precise physical and biological models. Having now established a general method for optimizing the efficiency and effectiveness of experiments, the ARCO algorithm can be applied to additional experimental parameters, forming a critical component of testing and validating a hypothesis.

I applied and evaluated ARCO for three reasons: **first**, to minimize a technical parameter such as light exposure (or sampling and refocusing rate), and to reduce phototoxicity and photobleaching; **second**, to select an optimal algorithm to evaluate the acquired data. And **third**, to combine both approaches and to provide an *overall system optimization.* Further concrete imaging issues were addressed, as was shown in this thesis in chapter 2 and chapter 4. Modifying the measuring point, based on a model of the phenotypes of interest, enables ARCO to maximize the amount of relevant cells needed in the pre-scanning step. The main hypothesis motivating this research was:

> *"Models of biological systems can be used in automated acquisition and analysis systems to optimize parameters of scientific experiments. Automated systems thus enhance information density and content."*

In short, this thesis confirmed the hypothesis above and found that the Automated Result / Cost Optimization algorithm implemented in LifeXplorer provides a powerful and adaptable tool for optimizing data acquisition and analysis during runtime. Furthermore, the thesis demonstrated that ARCO can be applied to other experimental parameters, such as focusing algorithms, sampling time, stacking, resolution and magnification. Future algorithms can be annotated in runtime to explain which experiment specifications worked best with which parameter-sets. ARCO and LifeXplorer finally can lead to an evolutionary network, which supports imaging and data analysis optimization based on subjective scientific questions and experiment specifications.

5.2 Future work

5.2.1 Genome In A Day

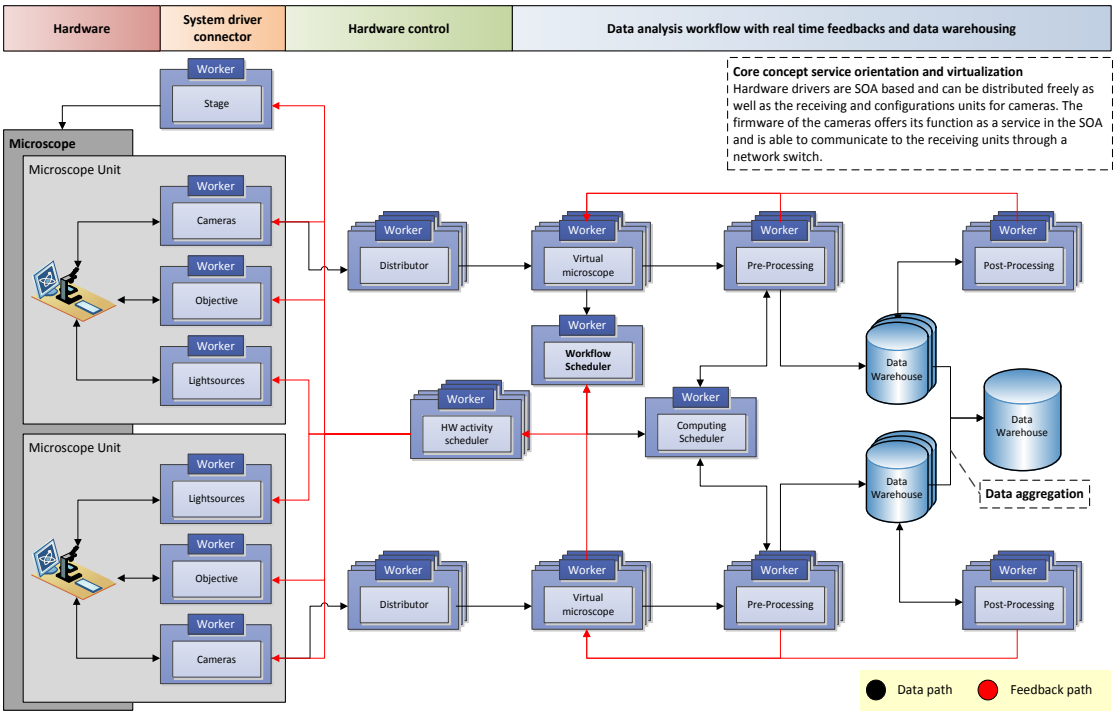


Figure 76 Genome In A Day Microscopy System Software Architecture

"Genome in a day" is the challenge to sequence (and analyze) the whole genome in one day. The challenge also envisions imaging the whole genome with one microscope in one day, which, however, far exceeds the capacity of current technologies. "Genome in a day" implies rather the employment of multiple lenses in the sequencing of the whole genome.

A number of technological advancements have been made that make plausible the sequencing of an entire genome in one day. Firstly, commercially available sequencers have vastly increased processing speeds; what used to take over a year can now be done in a matter of weeks (sequencing and analysis), a

rate of improvement that suggests ever-increasing speeds. Secondly, the costs of sequencing a whole genome have markedly been reduced, from $2.7 billion (1990 till 2010) to $1.000 (2004). Because of these two points, this study believes that sequencing and analyzing a genome in one day will soon become a reality.

Today, the methods involved in researching biological systems are called high throughput and next-generation sequencing methods. Modern sequencers have been used to produce the large quantities of data located at storage facilities in the International Cancer Genome Consortium (ICGC). “The ICGC was launched to coordinate large-scale cancer genome studies in tumors from 50 different cancer types and/or subtypes that are of clinical and societal importance across the globe. Systematic studies of more than 25,000 cancer genomes at the genomic, epigenomic and transcriptomic levels will reveal the repertoire of oncogenic mutations, uncover traces of the mutagenic influences, define clinically relevant subtypes for prognosis and therapeutic management, and enable the development of new cancer therapies.” [102]. A few articles on Medulloblastoma, a fast and aggressively growing tumor, have been published, stemming from research undertaken at the ICGC. [156, 157]. Their approach in analyzing the tumor is based on parallelizing the read out, an approach that has already been successfully applied in other contexts. The technique of parallelizing the measurement has been deployed in high-throuput microscopy. But it has not yet been possible to create successfully a parallel microscope consisting of multiple parallel microscopy units. Optical channels are split into different light paths; this enables the illumination to be triggered successively and as fast as possible for each channel. For future applications in microscopy, however, live cell imaging will probably play a crucial role in gaining deeper insights into biological systems. To measure the whole genome even time resolved, however, implies parallelism of the observation units. Parallelism, on the other hand, provides a limited approach, because it is not able to reduce complexity. It enables a much faster read out but causes a huge variety of unwanted issues in the data processing path. This thesis shows that real-time data selectivity increases information density and makes more specific observations. By means of ARCO, imaging can be made much more intelligently and its application can make

experiments more specific and relevant. Combining hardware parallelism and intelligence, however, addresses both speed and information densification.

In addition to generating a fruitful understanding of scientific automation, this thesis aimed to research the effects of parallel microscopy on feedback-controlled microscopy. The thesis detailed the architecture generally used to parallelize data processing and feedback control units. **Figure 76** depicts a framework for concretely handling the massive amounts of parallel microscopy units' parallel data. It shows that workers abstract a processing node specialized in processing and various management tasks. After the microscopy units are distributed, workers are able to flexibly send data to the processing network based on workload balancing techniques. A separate virtual microscopy unit exits for each physically existing microscope unit. A workflow scheduler controls the hardware and the computing workflow. The data processing is split into pre- and postprocessing, while the data warehousing is split into an aggregation hierarchy. First experiments indicated the importance of hierarchal data centrality. This is because data management systems running on a single node or multiple CPUs are not able to handle large numbers of objects in parallel. Database mangement systems, like the Microsoft SQL Server, are generally designed and optimized to provide high performance data handling. ARCO applications, however, may depend on single particle information for multiple measuring positions, which can even be an overload for a single instance database management system. Because of this need they will have to be mapped on this hierarchical data-warehousing design. As a first step, a database management system will be needed to handle each measuring position. In a second step, all information of one experiment should be aggregated, including information that exist in multiple measuring positions. And, finally, all information can be stored in one database to create final statistics and models. The post-processing path, however, is designed here to be based on lower hierarchy levels, not on the highest levels of aggregation which includes the information of all experiments and measuring points. Pre- and postprocessing generate feedback the virtual microscope unit, which, if necessary, can then add or change tasks in the global workflow scheduler. The scheduler distributes the tasks either to the computing or hardware acitivity scheduler. The

said design implies specifically that the microscopy units require eithernet ports for sending data from their physical location to the distribution workers. The microscopy units also need to have streaming functionalities to allow the imaging to send data without additional pull-mode communications from the distribution of workers. This architectural design of the units is necessary to avoid interrupt overflowing on the software layers of the workers' distributions, which can be caused if too many cameras send data to the same distribution node. At the same time, workload balancing and redundancy have to be possible, which means that cameras operating during runtime can be configured to send their data to different distribution workers. The architecture illustrated in **Figure 76** offers a preliminary plan for integrating intelligent, ARCO-based algorithms with parallel microscopy.

5.2.2 LifeXplorer Open Network (LEON) – evolutionary ARCO applications run on a shared computing network

The success of life sciences depends on its ability to handle complexity. Two promising approaches lend themselves best to the successful management of complexity. The first approach is reducing complexity as much as possible. The second is managing complexity by means of automation. I suggest that the approaches are not mutually exclusive; they ought to be combined. The success of the first approach rests on the quality of the scientific questions being posed. Scientific questions have to be intelligent and focused. In combination with their answers they have to build a resilient block for others which they can work with. Both approaches involve the employment of stable methodologies to make even small scientific answers reproducible. Handling the complexity of manageable scientific questions in the case of quantitative biology is still a huge challenge. This is because the insight generation workflow is long and complex, independent from the complexity of the scientific question at hand. In addition, there are a wide variety of scientific questions being asked at this stage of the young field of quantitative biology, too much to be answered at once. Even very focused and intelligent scientific questions will bring about complexity. Even if the first approach of reducing complexity is successfully performed, focusing on science automation [26] becomes increasingly relevant. Automation, however, already exists. From observation to mathematical models, automation is a crucial and essential part of modern research. At the same time, automation cannot yet significantly increase the reproducibility of scientific results. The main reason for this could be that even if automation is implemented, it is still heavily bound to expert knowledge and single persons or groups. This means that the accessibility of existing, mathematically formulated knowledge - including ways to extract image features, describe a phenotype, and make reliable statistics - is still exclusive and local and often not generalized. The knowledge, however, to handle a large variety of scientific questions exists in principle, but is distributed and often not accessible by computers.

Automation comes into play when researchers deal with issues of accessibility and the generalization of knowledge. A lab will greatly benefit from understanding a certain biological system. Research becomes easier and straight forward. Digitalization and world-wide knowledge sharing is key to reaching higher levels of efficiency in the life sciences. Classifying and relating data to each other creates knowledge. I therefore propose a network in which algorithmic knowledge about experiments and the workflow of answering scientific questions can be shared and easily used without the need for expert knowledge. The working title of this project, to create such a network of knowledge and models is: LifeXplorer Open Network (LEON). The idea behind LEON is that biologist and other scientist literally ask scientific questions. LifeXplorer connects then with LEON, asking for possible solutions to scientific enquiries. LEON loads and communicates the relevant models of the measured quantities and the process models, feature extraction and feedback control algorithms. This entire process is illustrated in **Figure 77**.

Laboratory instruments, such as a microscope, are prepared and the samples are created automatically in the future. After measuring the relevant information and analyzing it with the algorithms, which were found to be suitable to answer the initial question, the scientist then selects the result which he or she thinks is most realistic and suitable. LEON, at the same time, learns which algorithms work best in which situation. Eventually, this interactivity between operator and LEON leads to an evolutionary learning of LEON. Any new data analysis or model can be added, if the existing solutions are not suitable or no solutions exist in the network. By adding both models and algorithms as well as assigning experiment specifications with suitable measuring and analysis workflows, scientists automatically and instantly benefit from their colleagues all over the world. Certainly not everyone will be interested in such an approach, as it is based on openness and knowledge sharing. Participants of this network, however, will boost their efficiency. Others will be motivated to join, benefit from, and give something back to the network.

The main features of intelligent microscopy and more generally intelligent experiments will be based on the ARCO approach with an additional strategy

involved: to not only change the parameters of an experiment to optimize the result / costs ratio, but also verify results with LEON's knowledge. This additional function is a crucial enhancement and is necessary for the evolutionary approach of LEON. Further research, therefore, has to be performed on this topic.

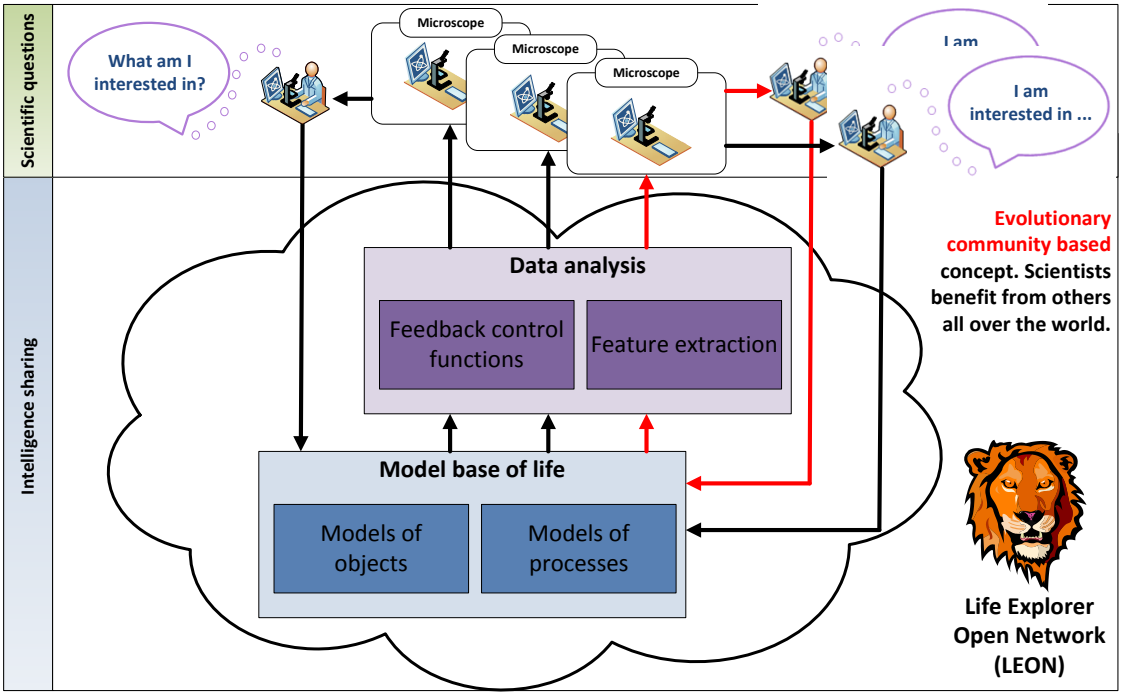


Figure 77 LifeXplorer Open Network (LEON) distributed architecture. Worldwide intelligence sharing of the systems biology community.

Figure 78 illustrates the LifeXplorer Intelligence Manager with several building blocks controlling essential imaging parameters. It additionally features real-time modeling. Each module here is an ARCO module that initially or even during runtime dynamically adapts to the needs of the experiment by testing different parameters and different data analysis algorithms. Preliminary evaluations of this method demonstrate that the principle of this paradigm works not only for the automated selection of different autofocusing algorithms, but also for segmentation problems (s. chapter 2). Several algorithms for nuclei segmentation, Golgi complexes classification, particle detection and tracking were put into the

LifeXplorer Image Processing Library; they are managed by a class called ObjectDetectionManager. As its name reveals, this class serves as a manager class. Object detection algorithms can be registered through a proxy object called ObjectDetectionSystem, as can be seen in **Figure 79**. The manager pattern in this case already shows the principle of the LEON: it abstracts from the actual algorithms. If the scientific question is based on a quantity and needs object detection, like the number of cells in an image, the object detection manager can be executed by the ARCO algorithm with an additional loop that sweeps through all available object detection systems. It pinpoints which algorithm would most efficiently maximize the result / costs ratio. This approach, however evaluated in this thesis already, needs further research.

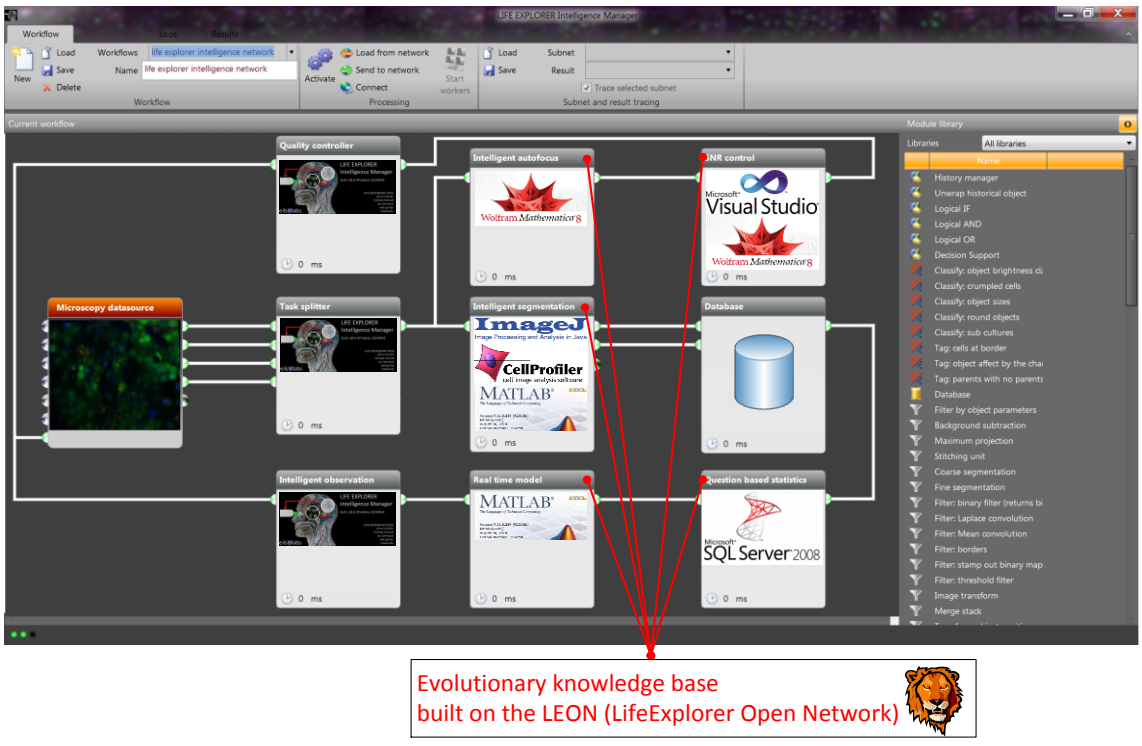


Figure 78 Evolutionary approach to abstract from the complexity of science automation

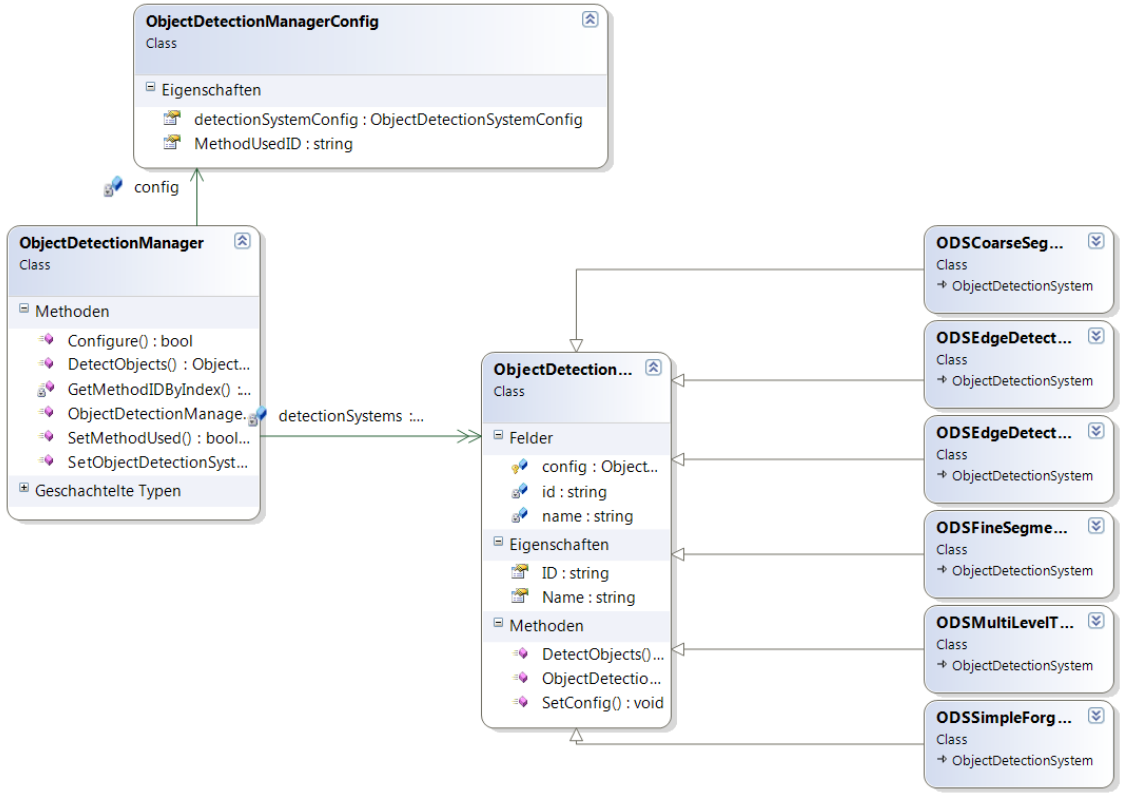


Figure 79 Class diagram of the ObjectDetectionManager class

Having explained and demonstrated the potential of ARCO by automatically selecting the most suitable algorithms for a specific scientific answer, making use of the LEON will now be explained. The abstract technical workflow in **Figure 80** is relevant for building ARCO into an evolutionary network that learns to answer scientific questions by intelligent measurements and data analysis algorithms that are shared and annotated by scientists all over the world. **Figure 80** depicts, in a sketch, relevant information in a lively and user friendly version that caters to the needs of a non-expert audience. It contains all important computing blocks when making use of LEON. The first step is to orally formulate a question. A speech to text algorithm then digitalizes the information. Once the experiment specification is digitized, a text mining algorithm extracts and relates previously known information in the experiment's specification. The operator, for example, might have explained that he or she is interested in general in autophagy dynamics and in more detail in cell nuclei and LC3 particles, and he or she wants to learn about the aggregation of particles over time at 16

measuring positions with a sampling rate of 30 minutes. The computer then extracts keywords like “LC3” and “particles” and brings them into relation. In this case, the ontology would be that LC3 is a subclass of particles, which is a subclass of a cell object, which is a sub class of an object. Furthermore, the imaging setup can be extracted.

Figure 80 First draft for operators of the LEON

By this human-machine interaction, the full experiment specification is defined and modeled. This experiment description model can be sent to LEON. Two steps are performed once the experiment description model is sent to LEON, as illustrated in **Figure 81**. Based on the experiment definition model, the static imaging or more general measuring workflow can be set up automatically. In parallel, the model can be sent to LEON for analysis. If, for example, information for the imaging workflow is missing, LEON can detect it based on existing knowledge in the network. A sample would be the missing information about the distance of the measuring positions. LEON can hold information about existing experiments that were also dealing with autophagy dynamics. This information

can then be loaded into the operator's graphical user interface, which must then be selected or confirmed. At the same time, existing data analysis algorithms and workflows, which were annotated to be suitable for the formulated scientific question, can be loaded. Initially, the network will not know how to extract LC3 particles, but it will then expand the search with the given ontology and find an object detection module which was annotated to detect objects in general. Behind this module, as explained above, the extended ARCO algorithm implements the Automated Algorithm Selection (AAS) algorithm. This algorithm depends on the definition of the measured quantities. For example, it may want to know, at least, the size of the objects. Again, this information had not yet been defined and the operator will be asked to input the necessary information. Once this information is inputted, however, LEON automatically annotates that LC3 particles have a specific size. To simplify the example, the magnification and resolution are not relevant, but, of course, both parameters will be additional information that LEON needs to know in reality. Returning to the annotation mechanism, LEON now knows basic information about LC3 particles and nuclei. The next operator, therefore, will find him or herself in a more comfortable situation, since the network has already learned how big LC3 particles are. Now, the imaging can begin and the loaded data analysis can work with the measured data. Once images are taken and sent to LEON's computing cloud, the general ARCO-based object detection module begins running. Having all the relevant information about the quantities that are to be extracted, like the size of the observed objects, ARCO can be executed for all existing object detection algorithms. It can exclude those which were not able to detect any object, and show all other results to the operator again, while the microscope is put on hold temporarily. The operator again provides crucial information to the LEON by selecting which result looks best. LEON annotates the selected data analysis suitable for the experiment definition. The next time the identical experiment is run, a suitable data analysis is known. LEON had previously learned a great deal of information, i.e. that nuclei are not separable point sources of light, whereas LC3 particles are.

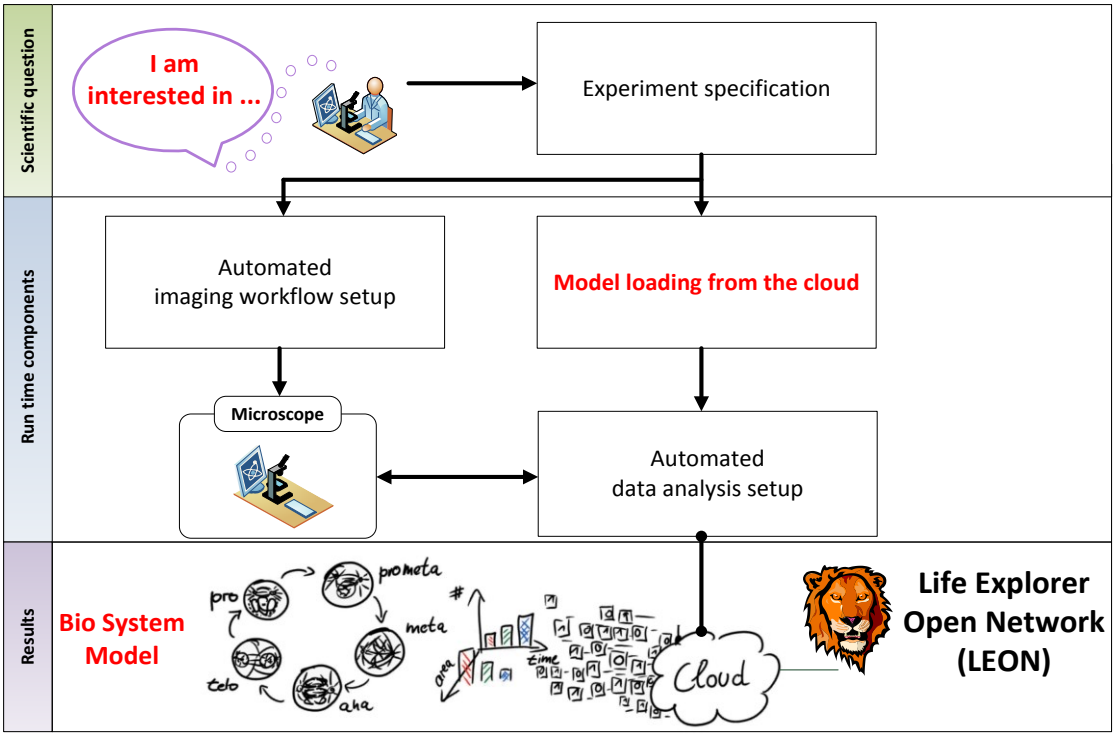


Figure 81 The workflow how to make use of the LEON

The moment another user has a similar question, the network will have multiple solutions and more knowledge about potentially suitable algorithms. It can again show the results to the operator. This time, the results can additionally receive rankings based on the existing usage statistics, such as which and how often keywords were used and for which algorithms, allowing the user to get a feeling for the community's decisions. Finally, the idea of LEON is very close to what is already state-of-the art in other daily-life applications like Google or Facebook. Ranking information by means of automation and user preferences is certainly the most basic swarm intelligence [158] approach. This approach, however, is very powerful once a user profile and context is given, which is the case for LEON and ARCO applications.

The methodology of ARCO and the LEON can be extended to statistics and modeling. In the case of statistics, which are supposed to be automated as well, like those illustrated in **Figure 81**, the same method of basic annotation and evolutionary learning can be used to provide statistical analysis packages to the

users based on the concrete needs formulated in the scientific question and experiment definition. The mathematical power of relating all steps from the question to the result with digital information should not be underestimated. Modern application networks have demonstrated how flexible and powerful the outcome can be, once information and solutions are created collaboratively. The automation of science by these means offer a necessary abstraction (and hide complexity) to biologists and other scientists, facing complex workflows, from their initial scientific question to the final answer. Organic computing and the ARCO algorithm, including existing technologies of information sharing and annotating, hardware control algorithms, and existing data analysis libraries can establish themselves into an open network for exploring life more specifically and easily than ever before.

Finally, ARCO, LifeXplorer, and LEON work together to increase both the efficiency and effectiveness of generating scientific knowledge, aiming to develop dependable therapies with broad application in the areas of health care, personalized medicine, and ontology.

Acknowledgements

This thesis work would not have been possible without being integrated with a fertile science environment and single individuals. Natural science is rationally approaching evolving life. Approaching to model life in a way it is reproducible in a controlled environment, and therefore becomes predictable. Being an apprentice of science, I had the good fortune to be supported especially by my advisors Roland Eils and Peter Fischer. Initially, I started working in the Viroquant project. Its technology platform was aiming to build an accelerated microscope, with which it would be possible to screen a whole genome in a day, and thereby win the "genome in a day" challenge. In my diploma thesis, however, I had the chance to do something nobody explicitly asked for: to make the microscope "intelligent" by integrating a general purpose feedback loop. Throughout this work, I had valuable and exciting discussions with Lars Lehmann and Peter Fischer. Having had a software company that created business intelligence solutions, I soon started to develop the idea to integrate the whole existing analysis pipeline and its data, with the purpose to be become more efficient and effective by a higher degree of automation. Peter Fischer "in contrast" motivated to become conscious about basic parameters of microscopy, such as the light exposure, bleaching and how this could be used to create an exposure control to stabilize the signal for image processing purposes. Combing both, my top down approach and the bottom up approach I had to learn, I started a journey with my PhD work towards LifeXplorer and the Automated Result / Cost Optimization (ARCO) algorithm which took me four years. All that only was possible because Roland Eils gave me the chance to continue the work on intelligent microscopy after my diploma thesis. From him I got valuable impulses to be focused on the biological application and to stay focused on the hard way to do something that has a certain impact. Roland gave me the high level support that was necessary for such a high risk project. When everything seemed to be lost because of a publication from the EMBL in 2011, Eric Manders, the inventor of CLEM (Controlled Light Exposure Microscopy), supported me in trusting my idea to use ARCO and the "model first" design for the light exposure

control presented in this thesis. In hard times, however, it was especially Nathan Brady and later also Ulrike Engel who supported me following my instincts and to implement and evaluate the methods which are presented in this thesis. I had great fun working together with Nathan Brady and also Stefan Maßen. Getting to know how biologists work today, what problems they are facing and also which methodology they use. It was valuable to see how the field itself developed throughout four years and how good we could perform, being outsiders with a fancy idea. I also am proud to be able to mention my little team, Michael Heinold and Axel Ganter. Both did their diploma thesis in the Viroquant project and were able to add value with their work. With Michael Heinold, who helped developing the Intelligence Manager, I could start further developments such as a framework for supervised classification. It was a good feeling that they trusted my leadership and understood how much responsibility I had to take over in my PhD work. I learned working together with strong personalities and high competences, however, at the time I was in commercial partnership with Thorben Burghardt, running Streamtec, our small enterprise for software development. A special person, who supported me emotionally, could feel my love for science and in some situations was the only person who could understand my emotions and to whom I could talk to about my worries was my dear friend Hans Raffée. The first time we meet, he immediately figured out what the potential of my approach can be. Although troubles in my PhD work were often too important to not be discussed, fortunately we also shared our thoughts about other fields of life and personal interests, such as religion and philosophy. For his wisdom and devotion helping a young apprentice of science, I am very thankful. Last but not least, I was supported by Claudia my wife, my family, Lara Krieg and Erich Schindler. They all helped me with patience and silent support. Especially Erich and also my father did a great job double checking my thesis for major grammar and spelling mistakes. Erich, even having a new born baby did not stop supporting me. Finally, I am thankful for every moment I could share with all the good and passionate scientists around the world and especially those being able to integrate both: the necessary (existing applications) and the possible (visions).

Figures

Figure 1 *The ARCO algorithm.*. *14*
Figure 2 *LifeXplorer architecture overview*..... *15*
Figure 3 *Schematic for the general three-step workflow of the ARCO algorithm*..... *22*
Figure 4 *Optimized time-lapse imaging using ARCO.*..... *26*
Figure 5 *General process of insight generation in science*..... *42*
Figure 6 *Compound microscope with two lenses*..... *48*
Figure 7 *Epi-illumination with Koehler illumination. Image adapted from [55].* *48*
Figure 8 *August Koehler introduced a new method of illumination*..... *49*
Figure 9 *The research light microscopy with inverted stand*..... *51*
Figure 10 *Multi-color imaging result sample*..... *53*
Figure 11 *Silicon as a photon-sensitive substrate in a CCD imager*..... *54*
Figure 12 *Digital image acquisition process*..... *55*
Figure 13 *Sequential assembly line*..... *57*
Figure 14 *Parallel assembly lines*..... *58*
Figure 15 *Basic service architecture*..... *66*
Figure 16 *Basic observer / controller architecture.*..... *68*
Figure 17 *Degree of self-organization*..... *70*
Figure 18 *.NET framework - language translation*..... *73*
Figure 19 *.NET framework version stack*..... *76*
Figure 20 *LifeXplorer software architecture.*..... *77*
Figure 21 *LifeXplorer: first architectural approach with data quality control*..... *78*
Figure 22 *LifeXplorer: first architectural approach with distributed computing*..... *78*
Figure 23 *LifeXplorer: data acquisition and processing architecture*..... *79*
Figure 24 *LifeXplorer main components and architecture.*..... *80*
Figure 25 *LifeXplorer implementation and application sample of a refocus trigger*..... *81*
Figure 26 *LifeXplorer abstraction layers.*..... *83*
Figure 27 *LifeXplorer basic module architecture*..... *84*
Figure 28 *LifeXplorer module samples*..... *85*
Figure 29 *LifeXplorer's Microscope Management System*..... *87*
Figure 30 *MMS worker resource information dialog*..... *88*
Figure 31 *Worker architecture*..... *90*
Figure 32 *Worker graphical user interface with run time information*..... *91*

Figure 33 *Worker task execution log* *92*
Figure 34 *LifeXplorer Intelligence Manager* *93*
Figure 35 *LifeXplorer ARCO circuit for intelligent light exposure control* *94*
Figure 36 *LifeXplorer's acquisition and processing framework in a feedback loop* *94*
Figure 37 *Organic Computing Manager architecture* *95*
Figure 38 *Organic computing: process distribution sample* *96*
Figure 39 *Organic computing scheduling strategies* *97*
Figure 40 *LifeXplorer's database design for time-resolved object statistics* *99*
Figure 41 *LifeXplorer's database design for parallel processing* *100*
Figure 42 *Relation between exposure time, information content and life time* *113*
Figure 43 *Golgi apparati segmentation efficiency for different exposure times.* *115*
Figure 44 *Signal minimizer.* *116*
Figure 45 *Approximates the necessary exposure time* *117*
Figure 46 *Controlled light exposure workflow* *118*
Figure 47 *Average intensity development with and without using ASEC* *119*
Figure 48 *Exposure time development* *120*
Figure 49 *ASEC with SNR control* *121*
Figure 50 *Exposure time development for ASEC with SNR control* *121*
Figure 51 *Mitochondrial energetics with and without applying ASEC* *123*
Figure 52 *Golgi apparatus dis- and reassembly over time* *125*
Figure 53 *Possible integration into existing Controlled Light Exposure* *126*
Figure 54 *"Possible autophagy-protein-dependent pathways* *128*
Figure 55 *Autophagosomes / LC3 development over time* *129*
Figure 56 *Lc3 particles over time with different exposure times* *132*
Figure 57 *Yeast membrane compartments* *133*
Figure 58 *Phototoxcity and photobleaching evaluation of ASEC* *137*
Figure 59 *(**a**) Model showing recycling of the Emp46–Emp47–Ssp120 complex* *138*
Figure 60 *Cross correlation of two different segmentation maps* *141*
Figure 61 *Phototoxicity and photobleaching evaluation of ASEC* *143*
Figure 62 *Components of a measured value.* *149*
Figure 63 *(**a**) Schematic representation of the microscopic preparation* *151*
Figure 64 *Cell nuclei of MCF7 cells.* *152*
Figure 65 *Surface modeling an acceleration workflow.* *153*
Figure 66 *Error histogram of focusing algorithms* *159*
Figure 67 *Histogram of all errors of all focusing algorithms for binned images.* *159*
Figure 68 *Acceleration of focusing by approximating the surface* *165*

Figure 69 *Cell nuclei of MCF7 cells* ... *175*
Figure 70 *Contrast quality over time.* ... *176*
Figure 71 *Refocusing trigger configuration form* ... *179*
Figure 72 *Different sampling points are computed in seperate subnets* ... *180*
Figure 73 *Refocusing trigger implementation in all LifeXplorer components* ... *182*
Figure 74 *Trigger distribution over time.* ... *183*
Figure 75 *Trigger latency distribution* ... *185*
Figure 76 *Genome In A Day Microscopy System Software Architecture* ... *190*
Figure 77 *LifeXplorer Open Network (LEON) distributed architecture* ... *196*
Figure 78 *Evolutionary approach to abstract from the complexity* ... *197*
Figure 79 *Class diagram of the ObjectDetectionManager class* ... *198*
Figure 80 *First draft for operators of the LEON* ... *199*
Figure 81 *The workflow how to make use of the LEON* ... *201*

Supplementary Figures

Suppl. Figure 1 *ARCO experiment workflow* ... *32*
Suppl. Figure 2 *ARCO application with LifeXplorer* ... *33*
Suppl. Figure 3 *ARCO Automated algorithm and parameter selection* ... *34*
Suppl. Figure 4 *Two separate application domains of ARCO* ... *35*
Suppl. Figure 5 *Experimental application of ARCO* ... *36*
Suppl. Figure 6 *Refocusing trigger system* ... *37*
Suppl. Figure 7 *Contrast quality over time* ... *38*

Tables

Table 1 *Levels of parallelism.* ... *56*
Table 2 *Automatic and manual overall ranking of 12 focusing algorithms* ... *156*
Table 3 *Precision statistics for focusing algorithms for all wells* ... *157*
Table 4 *Accuracy classes of focusing algorithms* ... *160*
Table 5 *Algorithm classes with binned data* ... *161*
Table 6 *Outlier detection statistics using the surface model* ... *164*
Table 7 *Accelerations factors based on standard deviations* ... *165*

Bibliography

1. Brazdilova, S.L. and M. Kozubek, *Information content analysis in automated microscopy imaging using an adaptive autofocus algorithm for multimodal functions.* J Microsc, 2009. **236**(3): p. 194-202.
2. Firestone, L., et al., *Comparison of Autofocus Methods for Automated Microscopy.* Cytometry, 1991. **12**(3): p. 195-206.
3. Sun, Y., S. Duthaler, and B.J. Nelson, *Autofocusing in computer microscopy: Selecting the optimal focus algorithm.* Microscopy Research and Technique, 2004. **65**(3): p. 139-149.
4. Conrad, C., et al., *Micropilot: automation of fluorescence microscopy-based imaging for systems biology.* Nature Methods, 2011. **8**(3): p. 246-U89.
5. Stephens, D.J. and V.J. Allan, *Light microscopy techniques for live cell Imaging.* Science, 2003. **300**(5616): p. 82-86.
6. Schneckenburger, H., et al., *Light exposure and cell viability in fluorescence microscopy.* Journal of Microscopy, 2012. **245**(3): p. 311-318.
7. Song, L.L., et al., *Influence of the triplet excited state on the photobleaching kinetics of fluorescein in microscopy.* Biophysical Journal, 1996. **70**(6): p. 2959-2968.
8. Song, L.L., et al., *Influence of fluorochrome labeling density on the photobleaching kinetics of fluorescein in microscopy.* Cytometry, 1997. **27**(3): p. 213-223.
9. Bernas, T., et al., *Minimizing photobleaching during confocal microscopy of fluorescent probes bound to chromatin: role of anoxia and photon flux (vol 215, pg 281, 2004).* Journal of Microscopy-Oxford, 2004. **216**: p. 197-197.
10. Hoebe, R.A., et al., *Controlled light-exposure microscopy reduces photobleaching and phototoxicity in fluorescence live-cell imaging.* Nature Biotechnology, 2007. **25**(2): p. 249-253.
11. Caarls, W., et al., *Minimizing light exposure with the programmable array microscope.* Journal of Microscopy, 2011. **241**(1): p. 101-110.
12. Zhong, Q., et al., *Unsupervised modeling of cell morphology dynamics for time-lapse microscopy.* Nature Methods, 2012. **9**(7): p. 711-3.
13. Popper, K.R. and W.W. Bartley, *Realism and the aim of science.* Postscript to the logic of scientific discovery1993, London ; New York: Routledge. xxxix, 420 p.

14. Sugden, J.K., *Photochemistry of dyes and fluorochromes used in biology and medicine: some physicochemical background and current applications.* Biotech Histochem, 2004. **79**(2): p. 71-90.
15. Edelstein-Keshet, L., *Mathematical models in biology.* Classics in applied mathematics2005, Philadelphia: Society for Industrial and Applied Mathematics. xliii, 586 p.
16. Eils, R., et al., *Quantitative imaging of pre-mRNA splicing factors in living cells.* Mol Biol Cell, 2000. **11**(2): p. 413-8.
17. Ellenberg, J., et al., *Nuclear membrane dynamics and reassembly in living cells: targeting of an inner nuclear membrane protein in interphase and mitosis.* J Cell Biol, 1997. **138**(6): p. 1193-206.
18. Gerlich, D., et al., *Four-dimensional imaging and quantitative reconstruction to analyse complex spatiotemporal processes in live cells.* Nat Cell Biol, 2001. **3**(9): p. 852-5.
19. Tvarusko, W., et al., *Time-resolved analysis and visualization of dynamic processes in living cells.* Proc Natl Acad Sci U S A, 1999. **96**(14): p. 7950-5.
20. Eils, R. and C. Athale, *Computational imaging in cell biology.* J Cell Biol, 2003. **161**(3): p. 477-81.
21. Snijder, B., et al., *Population context determines cell-to-cell variability in endocytosis and virus infection.* Nature, 2009. **461**(7263): p. 520-3.
22. Li, G.W. and X.S. Xie, *Central dogma at the single-molecule level in living cells.* Nature, 2011. **475**(7356): p. 308-15.
23. Snijder, B. and L. Pelkmans, *Origins of regulated cell-to-cell variability.* Nat Rev Mol Cell Biol, 2011. **12**(2): p. 119-25.
24. Snijder, B., et al., *Single-cell analysis of population context advances RNAi screening at multiple levels.* Mol Syst Biol, 2012. **8**: p. 579.
25. Jones, T.R., et al., *CellProfiler Analyst: data exploration and analysis software for complex image-based screens.* BMC Bioinformatics, 2008. **9**: p. 482.
26. King, R.D., et al., *The automation of science.* Science, 2009. **324**(5923): p. 85-9.
27. de Chaumont, F., et al., *Icy: an open bioimage informatics platform for extended reproducible research.* Nat Methods, 2012. **9**(7): p. 690-6.
28. Rusk, N., *Cell biology. Finding the trees in the forest.* Nature Methods, 2010. **7**(11): p. 869.
29. Erlich, Y., et al., *Alta-Cyclic: a self-optimizing base caller for next-generation sequencing.* Nature Methods, 2008. **5**(8): p. 679-82.
30. Krogan, N.J., et al., *Global landscape of protein complexes in the yeast Saccharomyces cerevisiae.* Nature, 2006. **440**(7084): p. 637-43.
31. Findler, N.V., *Machine Learning from Noisy Information.* Nature, 1964. **204**: p. 103.

32. Plaimas, K., et al., *Machine learning based analyses on metabolic networks supports high-throughput knockout screens.* BMC Syst Biol, 2008. **2**: p. 67.
33. Brady, N.R., et al., *Coordinated behavior of mitochondria in both space and time: a reactive oxygen species-activated wave of mitochondrial depolarization.* Biophysical Journal, 2004. **87**(3): p. 2022-34.
34. Zorov, D.B., et al., *Reactive oxygen species (ROS)-induced ROS release: a new phenomenon accompanying induction of the mitochondrial permeability transition in cardiac myocytes.* J Exp Med, 2000. **192**(7): p. 1001-14.
35. De Vos, W.H., et al., *Controlled Light Exposure Microscopy Reveals Dynamic Telomere Microterritories Throughout the Cell Cycle.* Cytometry Part A, 2009. **75A**(5): p. 428-439.
36. Hoebe, R.A., et al., *Quantitative determination of the reduction of phototoxicity and photobleaching by controlled light exposure microscopy.* Journal of Microscopy, 2008. **231**(1): p. 9-20.
37. Hoebe, R.A., C.J.F. Van Noorden, and E.M.M. Manders, *Noise effects and filtering in controlled light exposure microscopy.* Journal of Microscopy, 2010. **240**(3): p. 197-206.
38. Carpenter, A.E., et al., *CellProfiler: image analysis software for identifying and quantifying cell phenotypes.* Genome Biol, 2006. **7**(10): p. R100.
39. Kreshuk, A., et al., *Automated detection and segmentation of synaptic contacts in nearly isotropic serial electron microscopy images.* PLoS One, 2011. **6**(10): p. e24899.
40. Sbalzarini, I.F. and P. Koumoutsakos, *Feature point tracking and trajectory analysis for video imaging in cell biology.* J Struct Biol, 2005. **151**(2): p. 182-95.
41. Girish, V. and A. Vijayalakshmi, *Affordable image analysis using NIH Image/ImageJ.* Indian J Cancer, 2004. **41**(1): p. 47.
42. Wurtz, A., *Organic computing*, in *Understanding complex systems*2008, Springer: Berlin. p. xi, 355 p.
43. Brady, N.R., et al., *A wave of reactive oxygen species (ROS)-induced ROS release in a sea of excitable mitochondria.* Antioxid Redox Signal, 2006. **8**(9-10): p. 1651-65.
44. Shea, C.R., et al., *Rhodamine 123 phototoxicity in laser-irradiated MGH-U1 human carcinoma cells studied in vitro by electron microscopy and confocal laser scanning microscopy.* Cancer Res, 1990. **50**(13): p. 4167-72.
45. Carpenter, A.E., et al., *CellProfiler: image analysis software for identifying and quantifying cell phenotypes.* Genome Biology, 2006. **7**(10).
46. Brenner, J., et al., *An automated microscope for cytologic research.* J Histochem Cytochem, 1976. **24**: p. 100-111.

47. Conrad, C., et al., *Micropilot: automation of fluorescence microscopy-based imaging for systems biology.* Nat Methods, 2011. **8**(3): p. 246-9.
48. Kalkman, C.J., *LabVIEW: a software system for data acquisition, data analysis, and instrument control.* J Clin Monit, 1995. **11**(1): p. 51-8.
49. M., M., *MICROSCOPY APPARATUS*, 1961: USA.
50. M., M., *Memoir on Inventing the Confocal Scanning Microscope.* Scanning, 1988. **10**: p. 128-138.
51. Denk, W., J.H. Strickler, and W.W. Webb, *Two-photon laser scanning fluorescence microscopy.* Science, 1990. **248**(4951): p. 73-6.
52. Svoboda, K. and R. Yasuda, *Principles of two-photon excitation microscopy and its applications to neuroscience.* Neuron, 2006. **50**(6): p. 823-39.
53. Ratzlaff, E.H. and A. Grinvald, *A tandem-lens epifluorescence macroscope: hundred-fold brightness advantage for wide-field imaging.* J Neurosci Methods, 1991. **36**(2-3): p. 127-37.
54. Murphy, D.B. and M.W. Davidson, *Fundamentals of light microscopy and electronic imaging.* 2nd ed2013, Hoboken, N.J.: Wiley-Blackwell. p.
55. Janke, I., *Anwendung von Weitfeldmikroskopie mit strukturierter Beleuchtung für funktionelle Hirnabbildung*, 2004, Heidelberg: Heidelberg.
56. Tchan, Y.T., *Counting soil algae by direct fluorescence microscopy.* Nature, 1952. **170**(4321): p. 328-9.
57. Armstrong, J.A. and J.S. Niven, *Fluorescence microscopy in the study of nucleic acids; histochemical observations on cellular and virus nucleic acids.* Nature, 1957. **180**(4598): p. 1335-6.
58. Agard, D.A., et al., *Fluorescence microscopy in three dimensions.* Methods Cell Biol, 1989. **30**: p. 353-77.
59. Vrabioiu, A.M. and T.J. Mitchison, *Structural insights into yeast septin organization from polarized fluorescence microscopy.* Nature, 2006. **443**(7110): p. 466-9.
60. Marshall, W.F., et al., *Interphase chromosomes undergo constrained diffusional motion in living cells.* Curr Biol, 1997. **7**(12): p. 930-9.
61. Misteli, T., J.F. Caceres, and D.L. Spector, *The dynamics of a pre-mRNA splicing factor in living cells.* Nature, 1997. **387**(6632): p. 523-7.
62. Misteli, T. and D.L. Spector, *Applications of the green fluorescent protein in cell biology and biotechnology.* Nature Biotechnology, 1997. **15**(10): p. 961-4.
63. Takeuchi, K., et al., *Intracellular compartmentalization of fura-2 dye demonstrated by laser-excitation fluorescence microscopy: a problem in measuring cytosolic free calcium concentration using fura-2 fluorescence in vascular smooth muscle cells.* Tohoku J Exp Med, 1989. **159**(1): p. 23-35.

64. Ash, E.A. and G. Nicholls, *Super-resolution aperture scanning microscope.* Nature, 1972. **237**(5357): p. 510-2.
65. Hell, S.W., et al., *Confocal microscopy with an increased detection aperture: type-B 4Pi confocal microscopy.* Optics Letters, 1994. **19**(3): p. 222.
66. Hell, S.W. and J. Wichmann, *Breaking the diffraction resolution limit by stimulated emission: stimulated-emission-depletion fluorescence microscopy.* Optics Letters, 1994. **19**(11): p. 780-2.
67. Michaelis, J., et al., *Optical microscopy using a single-molecule light source.* Nature, 2000. **405**(6784): p. 325-8.
68. Willig, K.I., et al., *STED microscopy reveals that synaptotagmin remains clustered after synaptic vesicle exocytosis.* Nature, 2006. **440**(7086): p. 935-9.
69. Eggeling, C., et al., *Direct observation of the nanoscale dynamics of membrane lipids in a living cell.* Nature, 2009. **457**(7233): p. 1159-62.
70. Hell, S.W. and E. Rittweger, *Microscopy: Light from the dark.* Nature, 2009. **461**(7267): p. 1069-70.
71. Schroder, G.F., M. Levitt, and A.T. Brunger, *Super-resolution biomolecular crystallography with low-resolution data.* Nature, 2010. **464**(7292): p. 1218-22.
72. Aihara, K., et al., *An attempt to incorporate the automated tissue processing in electron microscopy.* J Electron Microsc (Tokyo), 1967. **16**(3): p. 285-7.
73. Rosen, D., *Automated microscopy.* Neuropathol Appl Neurobiol, 1981. **7**(5): p. 331-40.
74. Deindoerfer, F.H., et al., *Automated intelligent microscopy (AIM) and its potential application in the clinical laboratory.* Clin Chem, 1982. **28**(9): p. 1910-6.
75. Hiraoka, Y., J.W. Sedat, and D.A. Agard, *The use of a charge-coupled device for quantitative optical microscopy of biological structures.* Science, 1987. **238**(4823): p. 36-41.
76. Giloi, W., *Rechnerarchitektur.* Vol. 2. 1993, Heidelberg: Springer-Verlag Berlin Heidelberg.
77. Nof, S.Y., W.E. Wilhelm, and H.J. Warnecke, *Industrial assembly.* 1st ed1997, London ; New York: Chapman & Hall. xii, 500 p.
78. Nehmer, J. and Sturm Peter, *Systemsoftware*

Grundlagen moderner Betriebssysteme. 1. Aufl. ed1998, Heidelberg: dpunkt-Verl. IX, 362 S. graph. Darst.
79. Coulouris, G.F., *Distributed systems : concepts and design.* 5th ed2012, Boston: Addison-Wesley. xvi, 1047 p.

80. Kowalik, J.S. and R.M. Abarbanel, *High performance distributed computing: an introduction.* Stud Health Technol Inform, 2000. **79**: p. 187-94.
81. Buzbee, B.L. and D.H. Sharp, *Perspectives on supercomputing.* Science, 1985. **227**(4687): p. 591-7.
82. Orphanoudakis, S.C., *Supercomputing in medical imaging.* IEEE Eng Med Biol Mag, 1988. **7**(4): p. 16-20.
83. Cuticchia, A.J., *High performance computing and medical research.* CMAJ, 2000. **162**(8): p. 1148-9.
84. Kukkonen, C.A., *NASA high performance computing, communications, image processing, and data visualization-potential applications to medicine.* J Med Syst, 1995. **19**(3): p. 263-73.
85. Laxminarayan, S. and L. Michelson, *Perspectives in biomedical supercomputing.* IEEE Eng Med Biol Mag, 1988. **7**(4): p. 12-5.
86. Maizel, J.R., *Supercomputing in molecular biology: applications to sequence analysis.* IEEE Eng Med Biol Mag, 1988. **7**(4): p. 27-30.
87. Zhao, Y., et al., *A fast parallel clustering algorithm for molecular simulation trajectories.* J Comput Chem, 2012.
88. Bohm, K., *Supercomputing in cancer research.* Stud Health Technol Inform, 1997. **43 Pt A**: p. 104-8.
89. Bullard, D., et al., *Hydra: a self regenerating high performance computing grid for drug discovery.* J Chem Inf Model, 2008. **48**(4): p. 811-6.
90. Alerstam, E., T. Svensson, and S. Andersson-Engels, *Parallel computing with graphics processing units for high-speed Monte Carlo simulation of photon migration.* Journal of Biomedical Optics, 2008. **13**(6): p. 060504.
91. Zhang, K., Y.S. Wu, and G.S. Bodvarsson, *Parallel computing simulation of fluid flow in the unsaturated zone of Yucca Mountain, Nevada.* J Contam Hydrol, 2003. **62-63**: p. 381-99.
92. Cant, S., *High-performance computing in computational fluid dynamics: progress and challenges.* Philos Transact A Math Phys Eng Sci, 2002. **360**(1795): p. 1211-25.
93. Rao, A.R., G.A. Cecchi, and M. Magnasco, *High performance computing environment for multidimensional image analysis.* BMC Cell Biol, 2007. **8 Suppl 1**: p. S9.
94. Gärtel, U., W. Joppich, and A. Schüller, *First results with a parallelized 3D weather prediction code.* Parallel Computing, 1993. **19**(12): p. 1427-1429.
95. Bilbao-Castro, J.R., et al., *Exploiting desktop supercomputing for three-dimensional electron microscopy reconstructions using ART with blobs.* J Struct Biol, 2009. **165**(1): p. 19-26.

96. Adamson, B., et al., *A genome-wide homologous recombination screen identifies the RNA-binding protein RBMX as a component of the DNA-damage response.* Nat Cell Biol, 2012. **14**(3): p. 318-28.
97. Sheng, Z., et al., *A genome-wide RNA interference screen reveals an essential CREB3L2-ATF5-MCL1 survival pathway in malignant glioma with therapeutic implications.* Nat Med, 2010. **16**(6): p. 671-7.
98. Nybakken, K., et al., *A genome-wide RNA interference screen in Drosophila melanogaster cells for new components of the Hh signaling pathway.* Nat Genet, 2005. **37**(12): p. 1323-32.
99. Serviene, E., et al., *Genome-wide screen identifies host genes affecting viral RNA recombination.* Proc Natl Acad Sci U S A, 2005. **102**(30): p. 10545-50.
100. Cherry, S., et al., *Genome-wide RNAi screen reveals a specific sensitivity of IRES-containing RNA viruses to host translation inhibition.* Genes Dev, 2005. **19**(4): p. 445-52.
101. Liao, H.Y., M.L. Yin, and Y. Cheng, *A parallel implementation of the Smith-Waterman algorithm for massive sequences searching.* Conf Proc IEEE Eng Med Biol Soc, 2004. **4**: p. 2817-20.
102. Hudson, T.J., et al., *International network of cancer genome projects.* Nature, 2010. **464**(7291): p. 993-8.
103. Baird, J. and F. Turisco, *Service-oriented architecture: a critical technology component to support consumer-driven health care.* AHIP Cover, 2005. **46**(3): p. 74, 76, 78 passim.
104. Metsker, S.J., *Design patterns in C#*. The software patterns series2004, Boston: Addison-Wesley. xvii, 456 p.
105. Löwy, J., *Programming WCF services*. 1st ed2007, Sebastopol, CA: O'Reilly. xxii, 610 p.
106. Brown, P.C., *Implementing SOA : total architecture in practice*2008, Upper Saddle River, NJ: Addison-Wesley. xxx, 699 p.
107. Kitano, H., *Foundations of systems biology*2001, Cambridge, Mass.: MIT Press. xv, 297 p.
108. Branke, J.r., *Evolutionary optimization in dynamic environments*. Genetic algorithms and evolutionary computation2002, Boston: Kluwer Academic Publishers. xiv, 208 p.
109. Murch, R., *Autonomic computing*. On demand series2004, Upper Saddle River, N.J.: Prentice Hall Professional Technical Reference. xxiii, 309 p.
110. Arbib, M.A., *The handbook of brain theory and neural networks*. 2nd ed2003, Cambridge, Mass.: MIT Press. xvii, 1290 p.
111. Gittleman, A., *Computing with C# and the .NET Framework*. 2nd ed2012, Sudbury, Mass.: Jones & Bartlett Learning. xii, 753 p.

112. Wright, A., C.L. Hawkins, and M.J. Davies, *Singlet oxygen-mediated protein oxidation: evidence for the formation of reactive peroxides.* Redox Rep, 2000. **5**(2-3): p. 159-61.
113. Miyagawa, Y., M. Tamoi, and S. Shigeoka, *Evaluation of the defense system in chloroplasts to photooxidative stress caused by paraquat using transgenic tobacco plants expressing catalase from Escherichia coli.* Plant Cell Physiol, 2000. **41**(3): p. 311-20.
114. Bernas, T., et al., *Minimizing photobleaching during confocal microscopy of fluorescent probes bound to chromatin: role of anoxia and photon flux.* J Microsc, 2004. **215**(Pt 3): p. 281-96.
115. Sandison, D.R., et al., *Quantitative comparison of background rejection, signal-to-noise ratio, and resolution in confocal and full-field laser scanning microscopes.* Appl Opt, 1995. **34**(19): p. 3576-88.
116. Sandison, D.R. and W.W. Webb, *Background rejection and signal-to-noise optimization in confocal and alternative fluorescence microscopes.* Appl Opt, 1994. **33**(4): p. 603-15.
117. Santos, A., et al., *Evaluation of autofocus functions in molecular cytogenetic analysis.* J Microsc, 1997. **188**(Pt 3): p. 264-72.
118. Lee, M.C., et al., *Bi-directional protein transport between the ER and Golgi.* Annu Rev Cell Dev Biol, 2004. **20**: p. 87-123.
119. Klausner, R.D., J.G. Donaldson, and J. Lippincott-Schwartz, *Brefeldin A: insights into the control of membrane traffic and organelle structure.* J Cell Biol, 1992. **116**(5): p. 1071-80.
120. Lisauskas, T., et al., *Live-cell assays to identify regulators of ER-to-Golgi trafficking.* Traffic, 2012. **13**(3): p. 416-32.
121. Hoebe, R.A., et al., *Quantitative determination of the reduction of phototoxicity and photobleaching by controlled light exposure microscopy.* J Microsc, 2008. **231**(1): p. 9-20.
122. Kroemer, G., G. Marino, and B. Levine, *Autophagy and the integrated stress response.* Mol Cell, 2010. **40**(2): p. 280-93.
123. Saitoh, T. and S. Akira, *Regulation of innate immune responses by autophagy-related proteins.* J Cell Biol, 2010. **189**(6): p. 925-35.
124. Meyer, J.N. and A.S. Bess, *Involvement of autophagy and mitochondrial dynamics in determining the fate and effects of irreparable mitochondrial DNA damage.* Autophagy, 2012. **8**(12).
125. Levine, B., N. Mizushima, and H.W. Virgin, *Autophagy in immunity and inflammation.* Nature, 2011. **469**(7330): p. 323-35.
126. Andrzejak, M., M. Price, and D.H. Kessel, *Apoptotic and autophagic responses to photodynamic therapy in 1c1c7 murine hepatoma cells.* Autophagy, 2011. **7**(9): p. 979-84.

127. Ghavami, S., et al., *Apoptosis, autophagy and ER stress in mevalonate cascade inhibition-induced cell death of human atrial fibroblasts.* Cell Death Dis, 2012. **3**: p. e330.
128. Klionsky, D.J., et al., *Does bafilomycin A1 block the fusion of autophagosomes with lysosomes?* Autophagy, 2008. **4**(7): p. 849-950.
129. Hamacher-Brady, A., et al., *Artesunate activates mitochondrial apoptosis in breast cancer cells via iron-catalyzed lysosomal reactive oxygen species production.* J Biol Chem, 2011. **286**(8): p. 6587-601.
130. Jiang, Z., et al., *The role of the Golgi apparatus in oxidative stress: is this organelle less significant than mitochondria?* Free Radic Biol Med, 2011. **50**(8): p. 907-17.
131. Starkuviene, V., et al., *High-content screening microscopy identifies novel proteins with a putative role in secretory membrane traffic.* Genome Res, 2004. **14**(10A): p. 1948-56.
132. Conradt, B., *Cell biology: mitochondria shape up.* Nature, 2006. **443**(7112): p. 646-7.
133. Lin, M.T. and M.F. Beal, *Mitochondrial dysfunction and oxidative stress in neurodegenerative diseases.* Nature, 2006. **443**(7113): p. 787-95.
134. Fabene, P.F. and M. Bentivoglio, *1898-1998: Camillo Golgi and "the Golgi": one hundred years of terminological clones.* Brain Res Bull, 1998. **47**(3): p. 195-8.
135. Mironov, A.A., M. Pavelka, and SpringerLink (Online service), *The Golgi apparatus state of the art 110 years after Camillo Golgi's discovery*, 2008, Springer: Wien ; New York. p. viii, 716 p.
136. Babu, M., et al., *Interaction landscape of membrane-protein complexes in Saccharomyces cerevisiae.* Nature, 2012. **489**(7417): p. 585-9.
137. Donaldson, J.G. and J. Lippincott-Schwartz, *Sorting and signaling at the Golgi complex.* Cell, 2000. **101**(7): p. 693-6.
138. Melloni, E., et al., *In vitro photosensitizing properties of rhodamine 123 on different human tumor cell lines.* Photochem Photobiol, 1988. **48**(3): p. 311-4.
139. Groen, F.C., I.T. Young, and G. Ligthart, *A comparison of different focus functions for use in autofocus algorithms.* Cytometry, 1985. **6**(2): p. 81-91.
140. Liu, X.Y., W.H. Wang, and Y. Sun, *Dynamic evaluation of autofocusing for automated microscopic analysis of blood smear and pap smear.* J Microsc, 2007. **227**(Pt 1): p. 15-23.
141. Pengo, T., A. Munoz-Barrutia, and C. Ortiz-De-Solorzano, *Halton sampling for autofocus.* Journal of Microscopy-Oxford, 2009. **235**(1): p. 50-58.
142. Mateos-Perez, J.M., et al., *Comparative evaluation of autofocus algorithms for a real-time system for automatic detection of Mycobacterium tuberculosis.* Cytometry Part A, 2012. **81A**(3): p. 213-221.

143. Liu, X.Y., W.H. Wang, and Y. Sun, *Dynamic evaluation of autofocusing for automated microscopic analysis of blood smear and pap smear.* Journal of Microscopy-Oxford, 2007. **227**(1): p. 15-23.
144. Yazdanfar, S., et al., *Simple and robust image-based autofocusing for digital microscopy.* Optics Express, 2008. **16**(12): p. 8670-8677.
145. Bampton, E.T., et al., *The dynamics of autophagy visualized in live cells: from autophagosome formation to fusion with endo/lysosomes.* Autophagy, 2005. **1**(1): p. 23-36.
146. Wikstrom, J.D., G. Twig, and O.S. Shirihai, *What can mitochondrial heterogeneity tell us about mitochondrial dynamics and autophagy?* Int J Biochem Cell Biol, 2009. **41**(10): p. 1914-27.
147. Santos, A., et al., *Evaluation of autofocus functions in molecular cytogenetic analysis.* Journal of Microscopy-Oxford, 1997. **188**: p. 264-272.
148. Krotov, E., *Focusing.* Int J Comput Vis, 1987. **1**: p. 223:237.
149. Bueno-Ibarra, M.A., et al., *Fast autofocus algorithm for automated microscopes.* Optical Engineering, 2005. **44**(6).
150. Subbarao, M., T. Choi, and A. Nikzad, *Focusing Techniques.* Optical Engineering, 1993. **32**(11): p. 2824-2836.
151. Vollath, D., *The Influence of the Scene Parameters and of Noise on the Behavior of Automatic Focusing Algorithms.* Journal of Microscopy-Oxford, 1988. **151**: p. 133-146.
152. Netten, H., et al., *FISH and chips: Automation of fluorescent dot counting in interphase cell nuclei.* Cytometry, 1997. **28**(1): p. 1-10.
153. Brazdilova, S.L., *Towards reliable autofocusing in automated microscopy.* Icinco 2007: Proceedings of the Fourth International Conference on Informatics in Control, Automation and Robotics, Vol Icso, 2007: p. 440-447.
154. Zeder, M. and J. Pernthaler, *Multispot Live-Image Autofocusing for High-Throughput Microscopy of Fluorescently Stained Bacteria.* Cytometry Part A, 2009. **75A**(9): p. 781-788.
155. Lin, W., et al., *Automated local maximum-intensity projection with three-dimensional vessel tracking.* J Magn Reson Imaging, 1992. **2**(5): p. 519-26.
156. Rausch, T., et al., *Genome sequencing of pediatric medulloblastoma links catastrophic DNA rearrangements with TP53 mutations.* Cell, 2012. **148**(1-2): p. 59-71.
157. Jones, D.T., et al., *Dissecting the genomic complexity underlying medulloblastoma.* Nature, 2012. **488**(7409): p. 100-5.
158. Bonabeau, E. and C. Meyer, *Swarm intelligence. A whole new way to think about business.* Harv Bus Rev, 2001. **79**(5): p. 106-14, 165.