Department of Physics and Astronomy

University of Heidelberg

Master thesis

in Physics

submitted by

Frank Herrmannsdörfer

born in Stuttgart

2013

SimpleSTORM an Efficient Selfcalibrating Reconstruction Algorithm for Single and Multi-Channel Localisation Microscopy

This Master thesis has been carried out by Frank Herrmannsdörfer

at the

Interdisciplinary Center for Scientific Computing

under the supervision of

Prof. Dr. Fred A. Hamprecht

SimpleSTORM ein effizienter, sich selbst kalibrierender rekonstruktions Algorithmus für Einzel- und Mehrkanal-Lokalisationsmikroskopie:

Mikroskopie ist ein wichtiges Werkzeug der Zell-Biologie. Die Höchstauflösende STORM Mikroskopie gewährt genauere Einblicke in Zellen als andere Arten der Lichtmikroskopie, über die Abbe'sche Auflösungsgrenze hinaus. Dies wird durch die Aufnahme vieler Bilder mit jeweils wenigen und damit leicht voneinander trennbaren Bildpunkten erreicht.
SimpleSTORM ist ein effizienter Rekonstruktions-Algorithmus von STORM Daten, der von Schleicher (2011) entwickelt und im Rahmen dieser Masterarbeit weiterentwickelt wurde.
Für die Weiterentwicklung lag der Fokus, neben der Verbesserung des Algorithmus im Allgemeinen, auf der Vereinfachung der Bedienbarkeit im Besonderen. Durch Selbstkalibrierung ist die Angabe von Parametern wie dem Verstärkungsfaktor der Kamera oder der Breite der Punktantwort des Signals nicht mehr erforderlich. Diese werden automatisch bestimmt, können aber auch vom Benutzer eingegeben werden. Dies erleichtert auch unerfahrenen Benutzern die Arbeit mit SimpleSTORM.
Die Modellierung des Hintergrunds erlaubt es, Inhomogenitäten des Hintergrunds zu korrigieren und Aussagen über die Güte der Signale zu treffen.

Eine weitere Komponente der SimpleSTORM Software ist der Colorcomposer. Dieser ermöglicht die automatische Registrierung mehrerer Farbkanäle. Die Bestimmung der Kolokalisation zwischen verschiedenen Kanälen erlaubt die Untersuchung von Wechselwirkungen zwischen einzelnen Molekülen.

SimpleSTORM an efficient selfcalibrating reconstruction algorithm for single and multi-channel localisation microscopy :

Microscopy is an important tool in cell biology. Super resolution microscopy yields deeper insights into cells than other light microscopy methods, further than Abbe's resolution limit. This is achieved by taking many pictures with few and therefore easy separable fluorescent markers within each picture.
SimpleSTORM is an efficient algorithm which was originally developed by Schleicher (2011). The improvements of the general performance and usability will be described in this thesis.
Using selfcalibration it is no longer necessary to provide any parameter such as camera gain, camera offset or the expected signals point spread functions width. All crucial parameters are determined automatically but can also be manually adjusted. A model for the background is used to correct for inhomogeneous backgrounds, and to determine the quality of the signals found.

A second component of the SimpleSTORM software is the Colorcomposer. It provides easy auto alignment of different color-channels and measurements of colocalization, which is an important measure to investigate molecular interactions.

Contents

1 Introduction

Which proteins are involved in chemical synapses, how are they distributed? How and where in a cell do proteins interact? These questions and many more arise in biology and related sciences. Microscopy is a powerful tool to answer these questions, because it gives information about the shape and position of components of cells, the distribution of proteins and more. However light microscopy is limited in spatial resolution due to limited diffraction, as described by Abbe (1873) [1]. Different methods to increase the resolution of light microscopy beyond the diffraction limit have been developed recently. To name a few: Photoactivated Localization Microscopy (PALM) (Betzig et al. (2006)), Stochastic Optical Reconstruction Microscopy (STORM) (Rust (2006)) or Stimulated Emission Depletion STED (Hell and Wichmann (1994)).
For these techniques many images with sparsely distributed signals, coming from fluorophors, that are attached to the sample, are used. Special fluorophores are used that activate only with a low probability, which means that there are less fluorophores activated at a certain time, compared to other fluorescent microscopy. The pointlike fluorophores are blurred by diffraction. Because the signals are sparse the center of each signal can be determined with sub pixel accuracy. This is not possible for one image with all signals, where the blurred point spread functions overlap and are indistinguishable. There are different ways to stain the biological sample with fluorophors. One way is to use antibodies which bind to specific targets within a cell. The antibody is either directly linked to a fluorophore or two antibodies are used, where the first binds to the biological structure and the second with the fluorophore attached to the first antibody.
For this thesis only the dSTORM (direct STORM) method is considered. dSTORM is an advanced STORM technique which needs no activator fluorophore. It can be used to investigate the distribution of different proteins within a cell for example. To do so each kind of protein is labeled with a certain fluorophore. The images shown in this thesis were acquired to show signals from only one kind of fluorophore per channel. However, to be able to separate the different signals from the fluorophores, their emission spectrum must be distinct. This leads to chromatic aberration which results in images that are distorted and thus can not be aligned easily.
Many research groups have developed their own software to process STORM data sets. But most of these programs need many parameters, like a threshold for background suppression or the camera parameters, that must be set by the user to get reasonable results. Therefore these programs are difficult to use for somebody who is not familiar with image processing or does not know the parameters influence on the

[1] There is a fundamental limit for the resolution of optical systems. It is roughly half the wavelength of the light which is used for microscopy. With green light of 500 nm this means the minimal resolution achievable is about 250 nm

results.

One key question of this thesis is how to build software that is easy to use even without prior information about the data or without knowledge of image processing.

Our answer is SimpleSTORM, a software that calibrates itself. It estimates the camera parameters and the width of the point spread function of the fluorophores. In contrast to many other software applications, no threshold is needed. To provide this feature a robust background estimation is applied.

If more than one kind of fluorophore was used to stain the sample, the resulting images are distorted relative to each other due to chromatic aberration. The Colorcomposer tool was developed to do align multiple images automatically and to analyse the aligned images. Figure 1.1 shows both an reconstructed and aligned STORM image in the upper part and the maximum projection over all frames of the raw data in the lower part. The colocalization of the molecules of the different channels is a useful measure in biology to determine whether or not near molecules might interact. The Colorcomposer software calculates both global and local measures of localization.

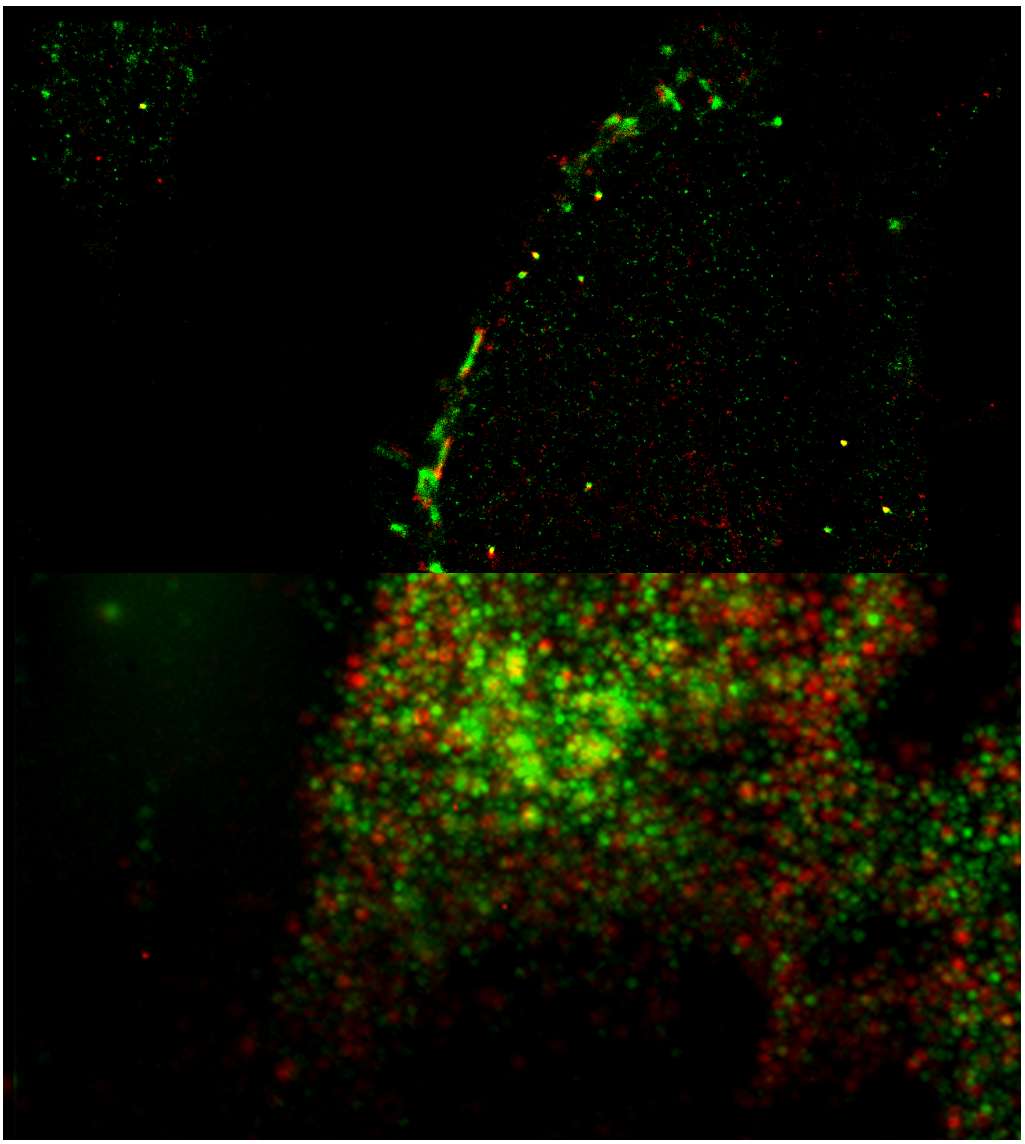


Figure 1.1: Dual-color image of a cell treated with nocodazole. The upper part shows the reconstructed and aligned image, the lower part the maximum projection of the raw data over all frames.

2 Theoretical background

This chapter gives an overview over the most important distributions and transformations that are used for the SimpleSTORM algorithm and the Colorcomposer. An introduction to the most important parts of regular Charge-coupled Device (CCD) cameras is also given, as well as a section about the data the algorithm is designed for.

2.1 The data

2.1.1 Labeling stuctures of interest

The concept of direct stochastic optical reconstruction microscopy (dSTORM) (Heilemann et al. (2008)) is, to label structures of interest with fluorophors that can be exited using a laser with the appropriate wavelength directly, in contrast to STORM techniques which need an additional activator fluorophore. Shortly after the activation the fluorophors emit a photon and fall back to the unexcited state or lose the ability to get exited, they bleach out. Special fluorophores have to be used that have a low probability to get activated. Each fluorophore is visible a couple of times at most. A movie that shows the frames consecutively shows blinking spots.
The technique to stain the samples that was used for all real-world images in this thesis is called immunofluorescence. For this technique antibodies are used to attach the fluorescenic molecules to the samples. Antibodies target specific biomolecules within a cell that show their antigen. There are two classes of immunofluorescence techniques.
Primary immunofluorescence uses only one antibody, which the fluorophore is directly attached to.
For secondary immunofluorescence two kinds of antibodies are used. First the primary antibody that binds to the molecule or structure of interest and then a second antibody that binds to the primary and carries the fluorophore. The primary antibody is extracted from a specific animal and is sensitive to the desired structure within a cell. The secondary antibody binds to any primary antibody from the specific animal (Fritschny and Härtig (2011)).
Using secundary immunofluorescence is more flexible because for different labels only different primary antibodies are needed, but the fluorophore that will be attached can be chosen freely. For primary immunofluorescence the combination is fixed.
The advantage of primary immunofluorescence is that the distance between fluorophore and target is smaller.
After staining the sample the unbound antibodies are washed away to avoid unspecific signals.

2.1.2 Description of the data sets

The datasets for dSTORM microscopy that we receive from our collaborators from Bioquant, Heidelberg are big datasets of several gigabyte in the Andor ".sif" format. Each file contains a stack of pictures, usually between 1000 and 10000, taken consecutively with a fixed exposure time normally between 20 and 200 milliseconds. The data set can be seen as a three-dimensional volume where the first two dimensions are given by the resolution of each picture and the thrid dimension is given by the size of the image stack. From now on the first two dimensions are called spatial dimensions. The third dimension of the data, resulting from the consecutive capturing of the images, is referred to as temporal dimension.

Figure 2.1 shows a typical frame of raw data. In each frame there might be multiple fluorophores visible at the same time. Due to the large magnification, beyond the diffraction limit, the almost pointlike fluorophores appear as approximately Gaussian shaped signals, their point spread functions. The fluorophores are either attached to the biological structures that are of interest or form a cluster, called a bead.

Beads are larger and brighter than spots from only one fluorophore and are used to align multiple channels in the postprocessing step. Designed to show up in every frame of the sequence at the same position they can be used as landmarks for alignment. They are composed of fluorophores of different colors to be visible in every channel.

The other spots, such as spots from fluorophores bound structures, often proteins of interest, only light up for a very short time. This is the key aspect of STORM. Instead of one frame that shows all fluorophores at the same time, thousands of frames are captured containing only a couple of point spread functions per frame. This makes it possible to determine the center of each point spread function with a sub-pixel precision. A high resolution image, with resolutions beyond the diffraction limit, can be reconstructed by plotting all detections found into one image.

2.2 Distributions

2.2.1 Gaussian distribution

The one dimensional Gaussian distribution is characterized by its mean μ and its variance σ^2 which is its standard deviation squared. The probability density function is:

$$f(x, \mu, \sigma) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right) \qquad \text{1-D} \qquad (2.1)$$

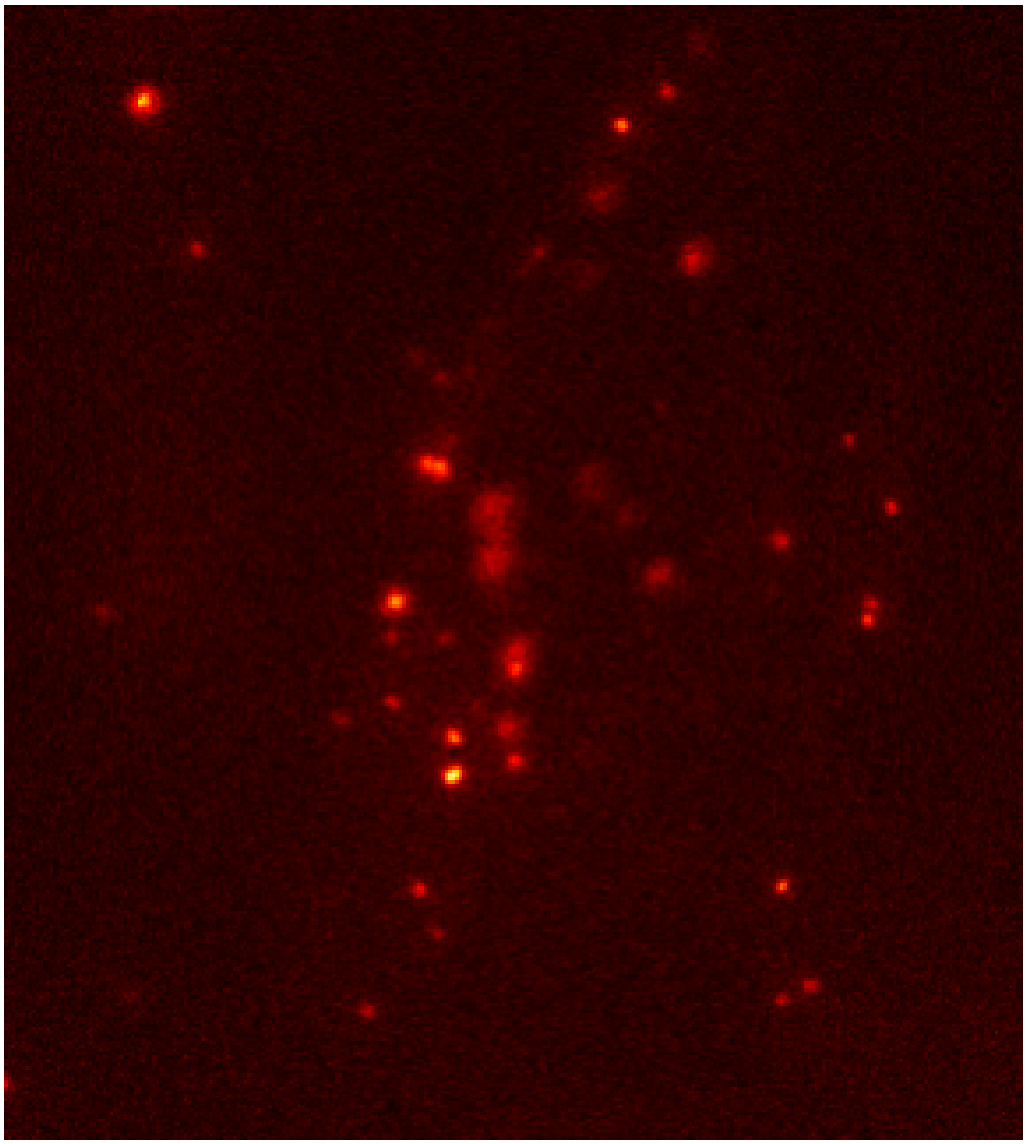


Figure 2.1: Raw image for dSTORM processing

The k-dimensional probability density function is characterized by the covariance matrixΣ and the is the k-dimensional mean $\boldsymbol{\mu}$:

$$f(x_1, ..., x_k, \boldsymbol{\mu}, \Sigma) = \frac{1}{\sqrt{(2\pi)^k \det(\Sigma)}} \exp\left(-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^{\mathbf{T}}\boldsymbol{\Sigma}^{-\mathbf{1}}(\mathbf{x}-\boldsymbol{\mu})\right) \quad k\text{-D} \tag{2.2}$$

Figure 2.2 shows some Gaussian distributions. The two dimensional Gaussian distribution is used for the Gaussian filter.

2.2.2 Poisson distribution

A very important probability distribution in Physics is the Poisson distribution. Its probability mass function is:

$$p(n, \mu) = \frac{\mu^n}{n!} \exp(-\mu) \tag{2.3}$$

The larger the mean value μ becomes the more likely the Poisson distribution results in a Gaussian distribution with a mean and a variance of μ. Figure 2.2 shows Poisson and Gaussian distributions with different parameters. The mean value of the Gaussian was shifted by 0.5 which results from continuity correction.
Poisson distributions describe the results of "counting experiments" and are therefore important for image processing as pictures taken with a camera are in principle counts of photons reaching the camera.
A Poisson distribution is defined for integer values only and its variance is the same as the mean value of the distribution.
The median of a Poisson distribution is approximately given by

$$\text{median Pois}(\lambda) \approx \lambda + \frac{1}{3} - \frac{1}{50\lambda} \tag{2.4}$$

Another important attribute is the skewness which describes the asymmetry of the distribution.

$$\text{skewness Pois}(\lambda) = \frac{1}{\sqrt{\lambda}} \tag{2.5}$$

The sum of Poisson-distributed independent variables is also Poisson-distributed.

2.2.3 Skellam distribution

The variance of a Poisson distribution can be estimated using a Skellam distribution. The probability mass function of the Skellam distribution is a function of the difference between two Poisson random variables

$$p(k; \mu_1, \mu_2) = \exp(-(\mu_1 + \mu_2)) \left(\frac{\mu_1}{\mu_2}\right)^{k/2} I_{|k|}\left(2\sqrt{\mu_1 \mu_2}\right) \tag{2.6}$$

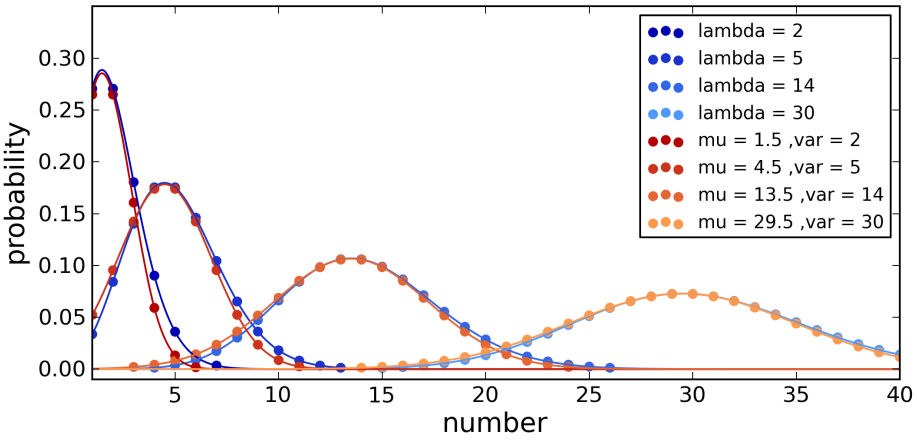


Figure 2.2: Poisson and Gaussian distributions with different parameters. Especially for small numbers the Poisson and the Gaussian distribution differ. The larger μ becomes for the Poisson distribution the more similar it becomes to a Gaussian with mean μ and sigma $\sqrt{\mu}$. The Poisson distributions were interpolated between their defined values. For better comparison the integer values of the Gaussian distribution were also marked with dots.

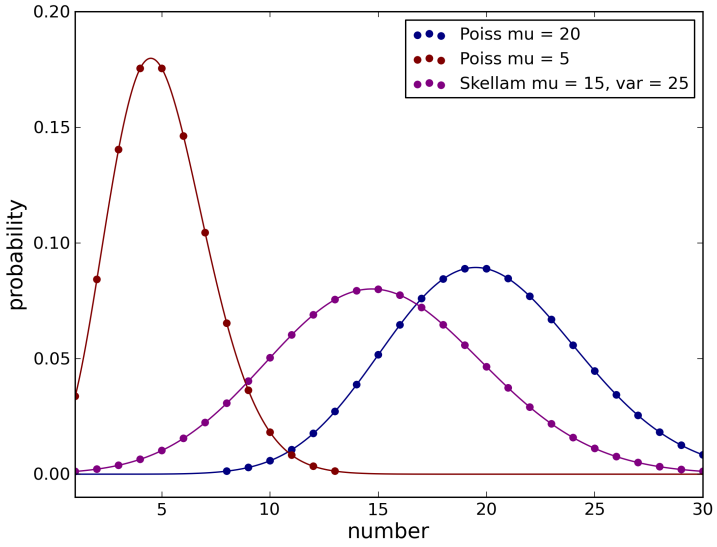


Figure 2.3: Two Poisson distributions (P_{red} and P_{blue}) and the resulting Skellam distribution. The distributions were interpolated between the integer values.

μ_1 and μ_2 are the means of two Poisson distributed random variables n_1 and n_2, $k = n_1 - n_2$. $I_{|k|}$ is the modified Bessel function of the first kind:

$$I_{|k|}(x) = \sum_{m=0}^{\infty} \frac{1}{m!\Gamma(m + |k| + 1)} \left(\frac{x}{2}\right)^{2m+|k|} \tag{2.7}$$

Mean μ and variance σ^2 of the Skellam distribution are given by

$$\mu = \mu_1 - \mu_2, \qquad \sigma^2 = \mu_1 + \mu_2 \tag{2.8}$$

$$\Rightarrow \qquad \mu_1 = \frac{\mu + \sigma^2}{2}, \qquad \mu_2 = \frac{-\mu + \sigma^2}{2} \tag{2.9}$$

Figure 2.3 shows two Poisson distributions (P_{red} and P_{blue}) and the resulting Skellam distribution (purple) of the difference $P_{\text{blue}} - P_{\text{red}}$. If the differences between following values of a single Poisson distribution are used as k for the Skellam distribution the variance of the Poisson distribution is given by the Skellam parameters μ_1 or μ_2 (the mean of the Skellam distribution μ will be close to zero if the same Poisson distributions are used).
In section 4.11 is shown which way to estimate the variance works better and more reliably.

2.3 Charge-coupled Device (CCD) camera

The light emitted by the fluorophores is captured by a CCD camera. During the capturing process the true measure, the photons arriving at the aperture of the camera, is transformed to electrons and eventually to a digital number. This section gives a short overview over the most important aspects of signal processing within a CCD camera.

2.3.1 Photon counting noise

The emission of photons is a random process that occurs at unpredictable times. Therefore the number of photons passing through a plane is never constant but varies around an average value. This means, that one can never determine exactly how many photons will hit the sensor chip of a CCD camera. This phenomenum is called photon counting noise. It plays a major role if the total number of photons is low, for example from dark sources or with short exposure times of the camera.

2.3.2 Quantum efficiency

Quantum efficiency describes the fraction of photons creating a detectable electron in a sensor chip. The quantum efficiency depends on the wavelength of the incoming photon. Photons with energies below the band gap can not produce a free electron that can be detected. The quantum efficiency has a maximum for a certain wavelength. This maximum of the quantum efficiency is basically caused by two effects. One hand the higher the energy of the photon is the higher is the kinetic energy of the freed electron and therefore the diffusion length is greater, on the other hand the electron is created further away from the detector and is more likely to recombine with an electron hole.

2.3.3 Gain factor

There are two different gain factors involved in the capturing process of a camera. Firstly the electric signal for each pixel may be amplified. Secondly there is a gain factor that describes the proportionality between collected electrons and the digital number it is associated with. The gain factor leads to a linear amplification of the input signal.

2.3.4 Readout noise

The origin of readout noise is the amplifier. The amplification is never perfect, this means that the exact number of electrons at the end of the amplification varies around the expected linearly increased value. There can also be random signals in the electronics adding to the signal. The readout noise does not depend on the exposure time.

2.3.5 Dark current noise

The dark current noise is generated by the thermal movement of the atoms in the sensor chip. The movement of molecules and atoms depends on the temperature of the material, therefore the dark current noise strongly depends on the chips temperature and can be reduced by cooling. Dark current noise generates electrons in the bins of each pixel, even with closed shutter. It increases constantly with time and follows Poisson statistics.

2.3.6 Quantization

The signal must fit into the output color depth. It has to be rounded or truncated to fit in. This process introduces errors that can be seen as additional noise, which depends on the intensity of the signal. High intensities are disturbed less in comparison to relative to low intensities.

2.4 Transformations

2.4.1 Transformation to Poisson distributed signal

The images acquired from the camera do not show the real intensities I_{true}, which result from the photon emission of the probe, but transformed intensities I_{meas}.
A linear transformation of the data is assumed with two parameters, the gain g and the offset o. Incoming photons create electrons via inner photoelectric effect. These electrons are collected for each pixel and might be amplified to get the final result. A linear relatiion between the number of incoming photons is assumed, as well as a linear amplifier. This assumptions lead to a linearly transformed signal with a slope of g. This factor is multiplied with the number of photons captured during exposure time for each pixel.
There might also be a constant offset for the pixel intensities that is not affected by the gain factor. If the gain factor g and the offset o are known the true intensity, the number of photons detected is:

$$I_{\text{true}} = \frac{I_{\text{meas}} - o}{g}. \tag{2.10}$$

After this transformation the intensities of the background are assumed to be Poisson distributed.

2.4.2 Anscombe transformation

The Anscombe transform (Anscombe (1948)) is used to transform a random variable with a Poisson distribution into one with an approximately constant standard

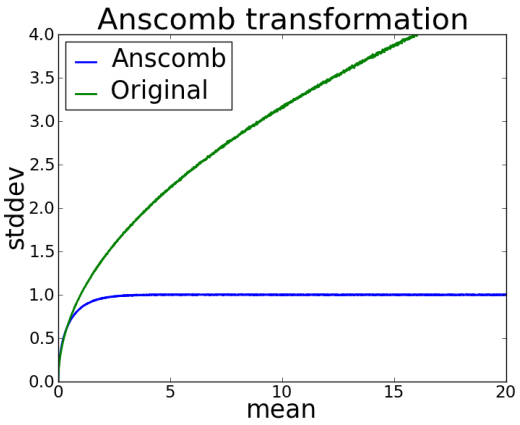


Figure 2.4: Standard deviation over mean intensities of different Poisson distributions (green) and their Anscombe transformed distributions (blue)

deviation. The transformation is defined as:

$$A(x) = 2\sqrt{x + \frac{3}{8}} + \frac{3}{8} \tag{2.11}$$

The inverse Anscombe transformation is given by:

$$I(x) = \left(\frac{x + \frac{3}{8}}{2}\right)^2 - \frac{3}{8} \tag{2.12}$$

Figure 2.4 shows the standard deviation of Poisson distributions with increasing mean (green line) and the standard deviation of Poisson distributions that are transformed useing the Anscombe transformation (blue line). As can be seen in Figure 2.4 the Anscombe transformations result has an intensity independent standard deviation of one for mean intensities greater than 4.

2.5 Estimation of camera gain and offset

Two different ways of estimating the camera parameters, which are needed to transform the background to a Poisson distributed background, have been investigated. For both approaches it is assumed that the intensities of the background and of the beads are Poisson distributed. This data I_{true} is then transformed in the following way:

$$I_{\text{meas}} = g \cdot I_{\text{true}} + o \tag{2.13}$$

This is the inverse transformation of equation 2.10.

2.5.1 Using variance-mean plot

The approach for the parameter estimation using the variances of the pixels is shown using a minimalistic example. It is then used in a generalized form.
Consider two pixels with different Poisson distributions P_1 and P_2 with mean value and variances of this distributions λ_1 and λ_2. If these distributions are transformed as given in equation 2.13 their mean and variance in temporal domain change as as follows:

$$\mathrm{var}(I_{\mathrm{meas1}}) = g^2 \cdot \mathrm{var}(I_{\mathrm{true1}}) \tag{2.14}$$
$$\mathrm{var}(I_{\mathrm{meas2}}) = g^2 \cdot \mathrm{var}(I_{\mathrm{true2}}) \tag{2.15}$$
$$\mathrm{mean}(I_{\mathrm{meas1}}) = g \cdot \mathrm{mean}(I_{\mathrm{true1}}) + o \tag{2.16}$$
$$\mathrm{mean}(I_{\mathrm{meas2}}) = g \cdot \mathrm{mean}(I_{\mathrm{true2}}) + o \tag{2.17}$$

This allows to determine the gain and offset using only two pixels. The values for $\mathrm{var}(I_{\mathrm{meas1}}), \mathrm{var}(I_{\mathrm{meas2}})$, $\mathrm{mean}(I_{\mathrm{true1}})$ and $\mathrm{mean}(I_{\mathrm{true2}})$ can be calculated from the data and can be used to calculate the gain as follows:

$$\frac{\mathrm{var}(I_{\mathrm{meas1}}) - \mathrm{var}(I_{\mathrm{meas2}})}{\mathrm{mean}(I_{\mathrm{meas}}1) - \mathrm{mean}(I_{\mathrm{meas2}})} = \frac{g^2 \cdot \lambda_1 - g^2 \cdot \lambda_2}{g \cdot \lambda_1 + o - (g \cdot \lambda_2 + o)} \tag{2.18}$$
$$= \frac{g^2 \cdot (\lambda_1 - \lambda_2)}{g \cdot (\lambda_1 - \lambda_2)} \tag{2.19}$$
$$= g \tag{2.20}$$

The offset o can be calculated in the following way:

$$\mathrm{mean}(I_{\mathrm{meas1}}) - \frac{\mathrm{var}(I_{\mathrm{meas1}})}{g} = g \cdot \lambda_1 + o - \frac{g^2 \cdot \lambda_1}{g} \tag{2.21}$$
$$= o \tag{2.22}$$

The estimation becomes more stable if more than two points are used and if the points are taken from a wide range of intensities. If more than two points are used a line of best fit must be found. This lines slope gives the gain factor and the intersection on the x-axis the offset. The variance of the pixels intensities are calculated directly, instead of using the Skellam approach. This is motivated in section 4.11.

2.5.2 Using skewness of Poisson distribution

A second approach is to use the skewness of the Poisson distribution to estimate the mean value.
For every pixel there are multiple intensities in temporal dimension. One can calculate

mean and variance of the measured intensities for each pixel $I_{\text{meas}}(i,j)$ and gets

$$\text{mean}(I_{\text{meas}}(i,j)) = g \cdot \text{mean}(I_{\text{true}}(i,j)) + o \tag{2.23}$$
$$\text{var}(I_{\text{meas}}(i,j)) = g^2 \cdot \text{var}(I_{\text{true}}(i,j)) \tag{2.24}$$

Assuming a Poisson distribution as the true intensity, mean and variance would be the same. Unfortunately the mean true intensities are unknown and it is not possible to determine g and o so far. For large mean Intensities μ the Poisson distribution becomes more and more symmetric and therefore similar to a Gaussian distribution with the same mean and variance as the Poisson distribution. However, for small means, the Poisson distribution is not symmetric. The skewness s_p of a Poisson distribution is the inverse of the square root of the mean $(\mu)^{-.5}$. It can also be directly calculated from data

$$s_p = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{x_i - \bar{x}}{\sigma}\right)^3 \tag{2.25}$$

The skewness is invariant to shift and multiplication with a constant. This means that the transformation caused by the camera gain and the offset does not affect the skewness. This gives a third equation to solve for g and o.
This approach is strictly limited to, at least for background pixels, low intensities. Tests have shown that if the mean of the true Poisson distribution is higher than roughly 30, the skewness gives no stable results. This is shown in section 4.9. For data sets with high background intensities it is not possible to determine the camera parameters in this way.

2.6 Check for correct gain factor

The gain factor may have been determined or entered incorrectly. Assuming there is the true gain factor g_{true} and the incorrect gain factor g_{false}. Let be $\text{Poiss}(\mu)$ the true Poisson distribution of one background pixel and o the offset. The pixels raw intensities are therefore $g_{\text{true}} \cdot \text{Poiss}(\mu) + o$. The following shows the effects of the incorrectly chosen gain factor on the transformed images:

$$\frac{(g_{\text{true}} \cdot \text{Poiss}(\mu) + o) - o}{g_{\text{false}}} = \underbrace{\frac{g_{\text{true}}}{g_{\text{false}}}}_{r}\text{Poiss}(\mu) \qquad \text{Transformation 2.4.1} \tag{2.26}$$

$$2\sqrt{r\text{Poiss}(\mu) + \frac{3}{8}} \approx \text{Gauss}(2\sqrt{r\mu}, r) \qquad \text{Transformation 2.4.2} \tag{2.27}$$

The variance of the Gaussian can be determined. Since the incorrect estimate of the gain factor and the Gaussians variance r are known, the correct gain factor can be determined by multiplication of the incorrect one with r.

2.7 Filters

Gaussian filter

The Gaussian filter uses a Gaussian for the convolution with the image that shall be filtered. The convolution of an image with a Gaussian filter alters the intensities of the pixels by adding some fraction, depending on the parameters of the filter, of the intensities of the surrounding pixels. The Gaussian filter can be used for noise suppression.

Wiener filter

The Wiener filter is also known as Wiener-Kolmogoroff filter. One application of the Wiener filter is noise suppression. It is designed to minimize the mean square error of the filtered image compared to the original image without noise.

3 The improved SimpleSTORM algorithm

STORM data from different cells or structures show many different features: there can be clusters of fluorophores with a high density of spots, areas with low density, variable background in space and time, beads can be present or absent.
This section describes how the algorithm processes the data sets and how it is possible to find good settings that will work for all kind of input data. The changes and improvements over the previous version are described and discussed. The new GUI design and the new features are introduced.
All real world images shown in this thesis were acquired from members of Prof. Dr. Mike Heilemanns group.

Import and processing

STORM data usually has a size of around 3 gigabytes. However larger data sets are possible too, making it necessary to work on smaller parts of the data, instead putting the whole dataset into memory. This is done using chunks of a user defined size. The data is processed chunkwise, and the processing of the frames of each chunk can be parallelized. This parallelization is possible because the signals in each frame are considered to be independent from each other. There is a dependency between different chunks. The mean values of several chunks are used to estimate the background. Therefore a certain number of mean values has to be stored in memory.

3.1 Workflow

3.1.1 Choosing parameters

Initially the user has the option to set all important parameters. If no parameter is set the default ones are used and will give a good result because all crucial parameters are either determined from the data or set to reasonable values that work for every data set. The goal of the default parameters is to give a good result with no adjustment. This means parameters are chosen to produce almost 100 % precision. It is assumed that the loss of some points that are not detected using this conservative setting will not affect the final result as much as a lower precision would.

3.1.2 Estimating camera gain and offset

First the application checks for a file containing settings for gain and offset from an earlier run. If this is not the case new parameters are estimated based on the first

part of the data; usually 200 frames are sufficient.
A set of 2000 pixels is chosen by their mean intensity with respect to time. Therefore the mean intensity range is divided in 2000 bins. For each bin one pixel with the appropriate mean is selected. For faster computation the mean is not calculated over all frames but only over a certain, user defined, number. The goal is to get pixels with the whole range of mean intensities, good representatives from the darkest background pixel to the brightest beads.
The method described in section 2.5.1 is used to estimate the gain factor, based on the preselected points. The variances of the pixels are computed based on the first part of the data. Each dataset is three-dimensional, where time is the third dimension. Therefore mean μ and variance σ^2, in time, can be calculated from the data for each pixel individually

$$\mu(i,j) = \frac{\Sigma_t(I_t(i,j)(i,j))}{n} \tag{3.1}$$

$$\sigma^2 = \frac{\Sigma_t(\mu(i,j) - (I_t(i,j)))^2}{n-1} \tag{3.2}$$

$I_t(i,j)$ describes the intensity of the pixel of frame t at position i,j. To determine the gain factor the variances σ^2 for each pixel are plotted over the mean intensities. A straight line can be fitted, and its slope gives the gain factor, the x-intersection gives the offset.
Figure 3.1 shows the scatter plot for the selected points and the fitted line. The data is taken from a real world data set. The red dots result from pixels that have a constant mean intensity over time. They are either dark the whole time, these are the dots on the left or they are beads and show signal in every frame. The blue dots result from pixels that are sometimes covered by a PSF. This blinking has much stronger influence on the variance than for the pixels mean values. This is the reason why this points are clustered on the lower end of the mean intensities.
For a robust estimation of the straight line a RANSAC (random sample consensus) algorithm is used. It is described in section 4.12.1.

3.1.3 Recursively adjusting gain and offset

After the estimation of gain factor and offset, the transformations described in section 2.4.1 and section 2.4.2 are applied and the background is subtracted.
Due to the Anscombe transformation, the background pixels of the image should only vary around a mean intensity of zero with a variance of one. Therefore, a histogram of the pixel intensities is created. After background subtraction, the background pixels should contribute only to the lower intensities in the histogram. A Gaussian function is fitted to the histogram's values. This is done under the assumptions that there is much more background in the image than signal, or that the intensities coming from signals are distributed over a larger range, so the Gaussian for the background intensity distribution can be fitted correctly.

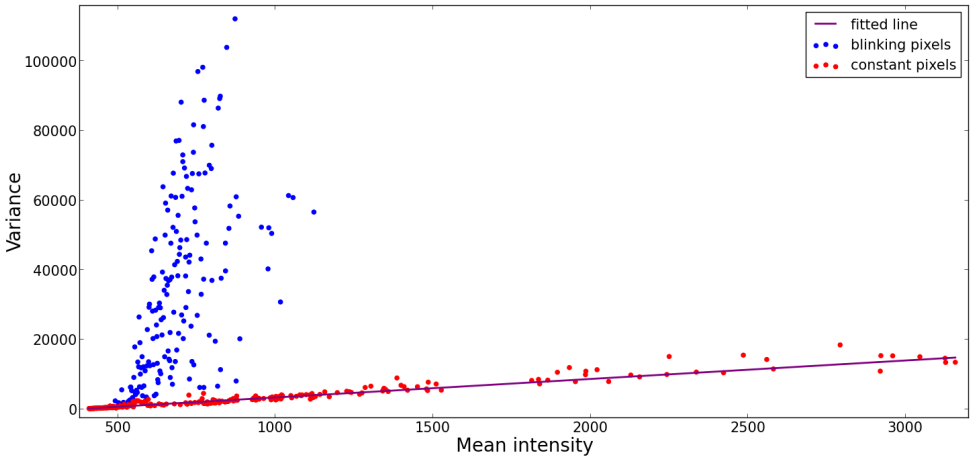


Figure 3.1: Scatter plot for the preselected points. Blue points result from pixels that show at least once a higher intensity caused by a fluorophore. The red dots are used to determine the gain and offset.

If the estimated value for the variance is too far from one, the originally estimated gain factor is corrected and applied, and the fit is done again (compare section 2.6). This is done until the background variance converges within a small threshold or the maximal number of iterations is reached. In this case the initial gain factor will be used and a warning will be shown on the screen.
It might be that the correction of the gain factor gets trapped duo to overshooting. In that case the estimated background variances oscillate around one. If this is the case the mean gain factor from the last two corrections is used.

3.1.4 Estimating the width of the point spread function

An inverse Anscombe transformation (equation 2.12) is applied to the data. This is necessary because the Anscombe transformation increases the point spread functions width, depending on the intensity of the point spread function. This is shown in section 4.6.
For 20 local maxima of a certain, user defined, number of frames the square of the Fourier transformation off each region of interest around the chosen local maxima is calculated. This squared Fourier transformations are then averaged. The result is called the mean power spectrum. It can be used to estimate the variance of the point spread function of the signal. A two dimensional Gaussian function's Fourier transform is again a Gaussian but with inverse variance. This relation is used to determine the variance of the point spread function in the spatial domain, using the fit parameter for the variance in the frequency domain.
The script for fitting the two dimensional Gaussian function was implemented by Ilia

Kats.

3.1.5 Processing the data

Import Data

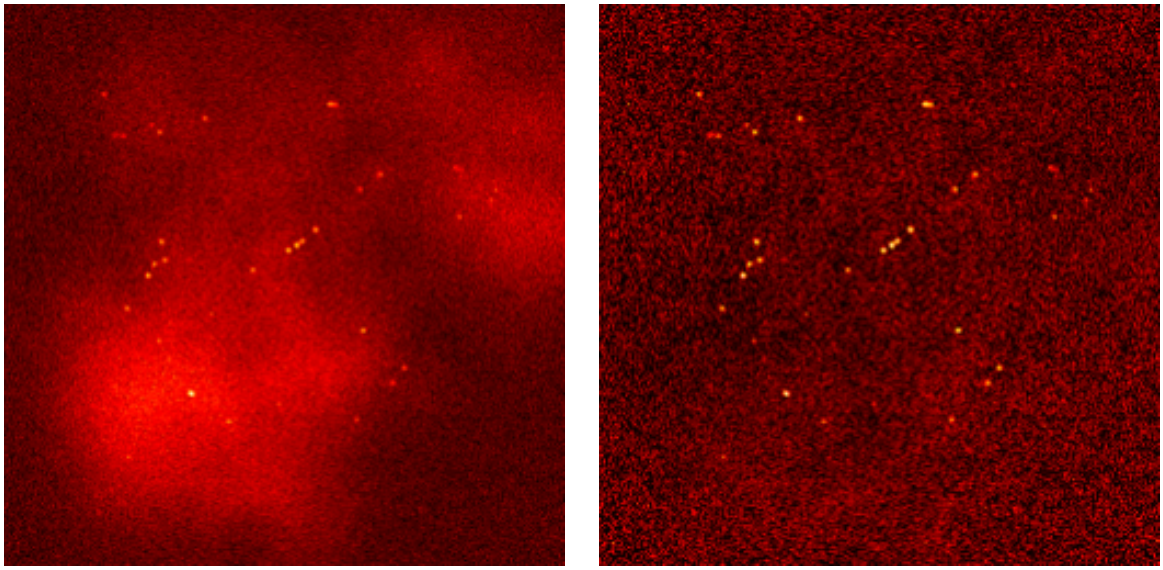
Storm data sets can consist of several thousand frames with resolutions up to one megapixel per frame. This size makes it necessary to break the data into smaller parts, otherwise it could be larger than the RAM of an ordinary machine. Because of the background estimation, it is not possible to process every frame completely independently as it was in the older version of this software (Schleicher (2011)).
This algorithm uses chunks of user defined size. There are some limitations to the chunk size that are discussed later. The data set is split into parts of equal size in the x- and y-dimensions and independently also in the t-dimension. If this partition does not fit at the edge of the data set, the last chunks will be smaller.
The data is transformed to be Poisson distributed, then the Anscombe transform is applied which gives background intensities with unit variance.
The implementation of the workflow using chunks was done by Ilia Kats.

Background estimation

For each chunk, the median of the Anscombe transformed data is determined to get a robust estimate of the background value for this chunk. B-spline interpolation, implemented in vigra (Köthe (2011)), is used to get interpolated values for the full resolution of the current frames. For this interpolation, three chunks in both the spatial and the temporal domain have to be available. Therefore the maximal chunk size into t-dimension must not be larger than a third of the total stack size, the same holds for the chunk size in x and y direction which must not exceed the spatial resolution of the image stack. These values are checked automatically and changed if necessary.
The interpolated background is then subtracted from the transformed data to give background pixels with zero mean and unit variance, both in xy- and in t-dimension. Figure 3.2 shows an frame with variable background before and after background subtraction.

Create mask for background suppression

With the given p-value from the settings, a global threshold can be determined, because inhomogeneities of background intensities have been removed. The threshold value is that intensity for which the integral over the probability density function of a Gaussian distribution with zero mean and unit variance gives $1 - p$. It is that intensity for which it is less likely than p to occur. This threshold is possible because the background intensities follow a Gaussian distribution, with known mean and variance, after all transformations applied.
The threshold is applied to the current frame and a binary mask is stored. For the



(a) Original image (b) image after background subtraction

Figure 3.2: Effect of background subtraction on an inhomogeneous background. For the human eye no more points are visible but for computers it is much easier to find the bright spots in the background subtracted image, because a global threshold can be applied or as in the case of SimpleSTORM the probabilistic model can be tested.

reason that it is possible for bright background pixels to exceed the threshold, the connected components of the mask are calculated. Pixels that belong to connected components with too few members are discarded. The idea behind using connected components is, that there is a probability for a background pixel to be brighter than the threshold, but it is unlikely that two neighboring pixels exceed the threshold in the same frame and even more unlikely that three of them do so. Therefore the number of pixels necessary for keeping a connected component is set to a minimum of three. If the width of the point spread function is very large the critical size of the connected components can be set depending on the PSF width. But to suppress background pixels efficiently the number of pixels in a connected component has to be larger than two.

Filtering data and finding maxima

To improve the accuracy of the spot detection, the transformed signal is convolved with a two dimensional Gaussian function with the previously determined or user-set width. The convolved image will further be used to find the maxima. All local maxima in the frame are detected. Each maximum found is tested to be covered by the mask or discarded otherwise. A region of interest around the remaining maxima is interpolated to a higher resolution. The interpolated region is searched for local maxima once again. These maxima will be detected with super resolution.

To determine the signal-to-noise ratio, the unfiltered and uninterpolated pixel intensity is used. This maxima detection was already implemented by Joachim Schleicher (Schleicher (2011)).

Quality control for detections

Sometimes, especially in data sets with a high density of spots, two spots are near enough that their point spread functions overlap. It may happen that instead of two maxima only one maximum will be detected between the true ones, as can be seen in Figure 3.3. This leads to large errors in the localization. To avoid this a threshold for the asymmetry of the spots can be set.
The calculation of the asymmetry was already implemented by Joachim Schleicher Schleicher (2011).

3.2 Comparison with older version of the SimpleStorm algorithm

3.2.1 Adjustable filter width

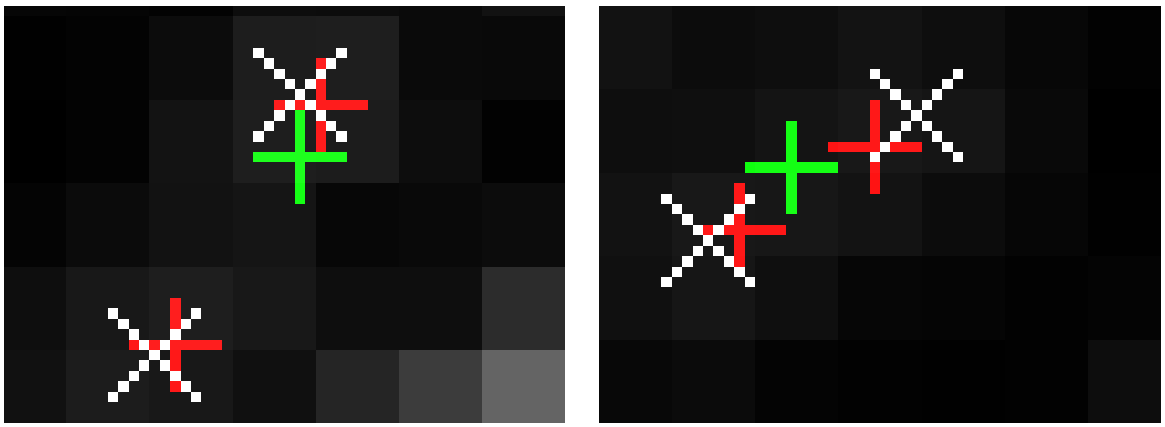
If two or more point spread functions (PSF) overlap, applying a smoothing filter can lead to the merging of point spread functions. This merging becomes a problem if the two distinct maxima of the original unsmoothed PSFs form a new maxima in between. This leads to just one detection somewhere between the true maxima. The number of merging PSFs increases with greater filter widths. For high density data it might be better to use a filter with smaller width and therefore less accuracy, but fewer merged PSFs. In total this might give a better result depending on the number of incorrectly merged PSFs. This is the reason why filtering was changed to use just a Gaussian filter instead of the Wiener filter originally used. The effect of the smaller filter width can be seen in Figure 3.3. The red crosses indicate detections found by the new SimpleSTORM version, the green crosses show the estimated positions found by the previous version of SimpleSTORM, and the white x marks the true location. The predictions by the newer version are not perfect but better than the predictions of the older version. The scores that can be seen in Table 3.1 are calculated like as described in section 7.3. Both results have almost the same accuracy, but differ in the scores. There are two effects related to the accuracy canceling each other out. Fewer merged PSFs increase the accuracy, but the positive effect of filtering with the appropriate filter, as shown in Figure 4.5 or in Figure 4.6, is lost.

3.2.2 False positive suppression

In the previous version the background was determined by first estimating a baseline. The minimum of the current frame was taken as the baseline. It was subtracted from the image and the result was smoothed with a Gaussian filter of width ten. The smoothed image was subtracted from the original image to give a background free

Table 3.1: Comparison of results created using almost no smoothing or Wiener filter on high density data. Although the accuracy is almost the same, the number of detections and the scores are higher for the unsmoothed data. For the evaluation software from the Biomedical Imaging Group (2013) were used.

	intersections	Jaccard	F-Score	Precision	Recall	RMSE
Gaussian filter, $\sigma = 0.01$	16955	19.99	33.26	81.10	20.92	27.21
Wiener filter	14480	17.06	29.15	79.19	17.87	27.28



(a) Advanced setting widget (b) Easy setting widget

Figure 3.3: This pictures show the effect of a smaller filter width. The red crosses show detections found with a filter width of 0.01, the green crosses the results using a Wiener filter. The white x marks the ground truth.

(a) Typical frame showing variable background intensities

(b) Result old SimpleSTORM

(c) Result new SimpleSTORM

Figure 3.4: This pictures show the drawback of the old background treatment. On the left a typical frame with variable background is shown. The old version of SimpleSTORM discards many detections in the regions of high background intensity. The new SimpleSTORM software discards detections based on their signal-to-noise ratio and is less affected by variable background.

image. This works fine for subtracting background with variations larger than the filters width, but the resulting intensities do not contain any information about the variability of the background intensities.
If, for example, the resulting intensity, after background subtraction, is five. This can either mean it is most likely signal, given that the variance of the background in the original image was very small, or it can mean nothing, if the variance of the original background pixels were 20. In the latter case the probability that the difference of five between the original image and the smoothed one has a high chance to result from the background variation.
In the older version of SimpleSTORM the intensity of a candidate, considered to be a signal, was checked whether its intensity after the background subtraction was higher than the estimated background value at the maximums position minus the baseline. This works for homogeneous background intensities but fails if the baseline, gets a small value from a region with small background values and is applied to a bright region. In that case the intensity of the estimated background gives high values at the maximums position but only a small value is subtracted. This results in discarding many maxima in regions of bright background as in Figure 3.4. The old version of SimpleSTORM only finds about 8000 spots while the new version finds more than 44000 on a test data set with high background variance. Using the new background suppression results in a reconstructed image that shows the reconstructed structures in a better way.

The great advantage of the newer version of SimpleStorm is its model for the background. For each pixel, the probability to be signal can be determined based on its signal-to-noise ratio. Using the number of connected components of each cluster, false positive detections caused by bright background pixels can be suppressed.

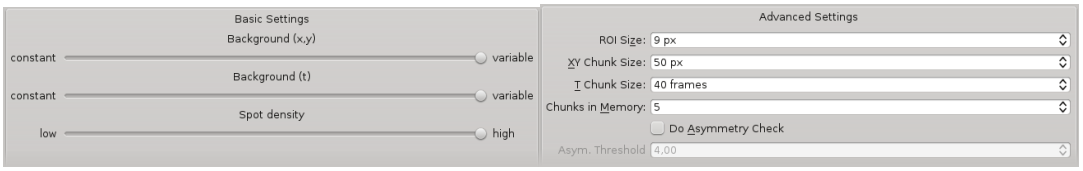
3.2.3 Comparable results based on the signal-to-noise ratio

The accuracy of the maxima detection relies on the signal-to-noise ratio (SNR). With a given SNR the correct standard deviation of the localization error can be used for further calculations (the calculation for the localization error is done in section 4.5). This is not possible if only intensities are saved because without information about the backgrounds variance the reliability of the detection can't be estimated.
Saving the signal-to-noise ratio also enables to compare results from either different cameras or different settings or a different environment that might lead to a higher background variability.

3.3 New graphical user interface (GUI)

3.3.1 Input widget

The new GUI for SimpleStorm was designed to integrate its many new features. Figure 3.6 shows this new design.
There are three categories of parameters. The first category specifies which upsampling factor will be used, the width of the pixels in nanometers of the input data, the reconstruction resolution, the number of frames that are used to estimate the camera parameters and the PSFs width and sensitivity of the algorithm. The first three parameters are general information that depend on the capturing process of the data and the desired upsampling factor. Two of them must be set and the third is calculated automatically.
The most challenging parameters of this section are the alpha value, which sets the sensitivity for false detections and the number of frames used for estimation of the camera parameters and the PSF. For these values the default setting are set to work well with any data sets.
The next category of parameters defines the width of the region of interest (ROI) for the estimation of the point spread function and the chunk sizes in spatial and temporal dimension can be set. There are two different ways to set these parameters (see Figure 3.5). One is to give values for all parameters. This is difficult without understanding the influence of these parameters on the algorithm. Therefore there is also a second way to set this parameters. The user should know some properties of the data that shall be processed, such as: is the spot density high or low? Is there variable background in time and space? Depending on the sliders' positions, the best parameters are set automatically. How this is done will be described in section 3.3.3. With this second option the user can process his or her data, treating variable background or dense data without deep insight or understanding of the algorithms.
The last category of parameters describes the camera gain and offset, the width of the signals point spread function, and a prefactor that can be used to alter the estimated gain.



(a) Easy setting widget (b) Advanced setting widget

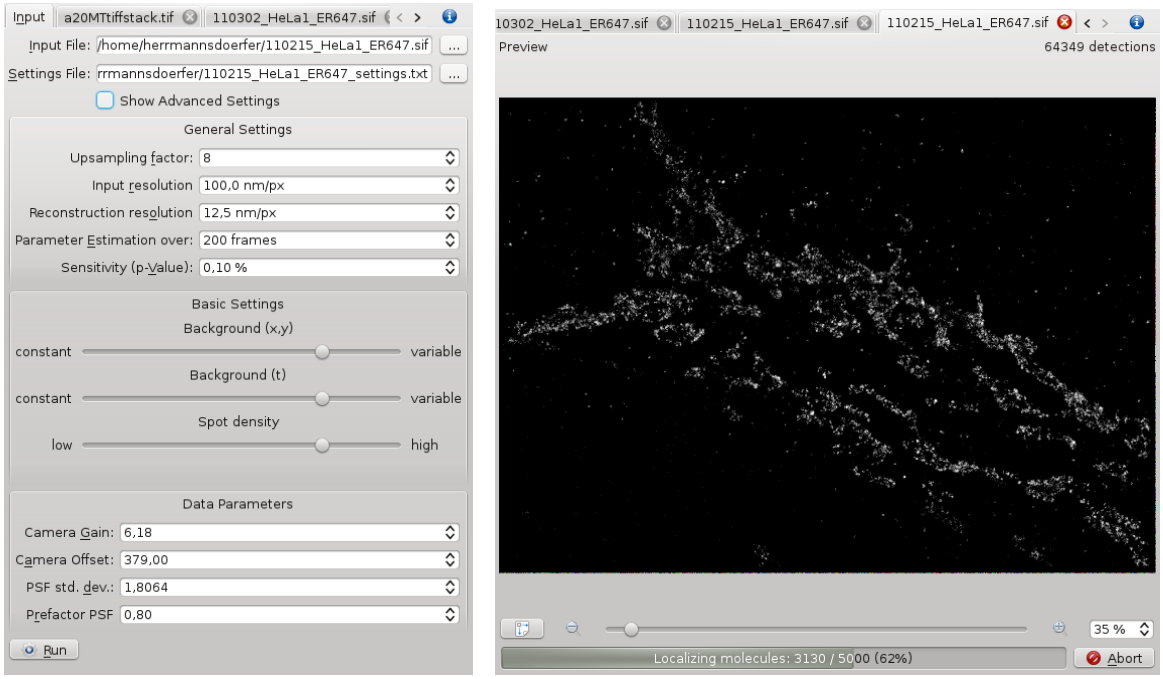
Figure 3.5: There are two different ways to set the parameters for the algorithm. On the right the standard way of setting parameters can be seen. On the left, there are sliders that can be adjusted between the extremes. The program sets the value for the parameters in a way to produce the best results for the selected attributes of the data set.

3.3.2 Result widget

After the run button at the lower left edge of the input widget is pressed, a new tab opens. In this widget the reconstructed image is shown. At the bottom there is a progress bar that displays the current processing step and its progress. Buttons to zoom in and out or to fit the displayed image into the window are located in the lower part of the result widget. On the lower right there is the stop/save button. It either stops the program if it is still running or opens a dialog to save the result image and the coordinates of the detection if the program has already been stopped.
The GUI was mainly designed by Ilia Kats.

3.3.3 Easy parameter selection

As mentioned above the easy setting widget makes it easy to set reasonable parameters without knowing their influence on the algorithms.
The background sliders have direct influence of the corresponding chunk size. A more constant background gives better results with larger chunks. This is because the median of all pixels in a chunk is used to estimate the mean value of the background. For the estimation of this relation between the median and the mean λ of a Poisson distribution, described in equation 2.4, is used. Because of the high mean values for λ the last summand can be dropped. This estimation works only if the chunks contain more background pixels than signal, otherwise the mean value will be overestimated. With large chunks this assumption is satisfied. But the smaller the chunk size, the more likely it is that a cluster of points lies in the chunk, and the values of the median are too high. On the other hand, the chunk sizes should be within the range of the changes in background. The best chunksize is therefore in the range of the variable background. The slider position sets the chunk sizes linearly to a value that lies in between the smallest and the largest chunk size possible for the data set.
The minimal possible chunk size is 3 pixels. The largest chunk size possible is half of the shorter border in x and y dimension and the number of frames over which the parameters are estimated divided by the number of chunks in memory in t dimension. These restrictions to the maximal chunk size result from the spline interpolation as



(a) Input widget (b) Result widget

Figure 3.6: The new GUI. On the left is the window for selecting input file and parameters. On the right is the result widget showing the processing of a data set in progress.

described in section 3.1.5.
In general, the denser the dataset is, the more likely it is that two neighboring PSFs are merged due to the filtering, as shown in section 3.2.1. There are two ways to avoid inaccurate detections that result from merged PSFs. One is less filtering, to avoid merges, the second one is checking the symmetry of the detected spots. Merged spots become more and more asymmetric the further the two true centers of the PSFs lie apart from each other. A high asymmetry is a good indicator for a detected spot with low accuracy. The spot density influences the prefactor for the estimated sigma and the value for the asymmetry checks. It sets the threshold for the asymmetry in the same way the sliders for background work, within appropriate limits, with higher values for the asymmetry threshold for less dense data sets. Also, the prefactor is set to values between 1 for sparse data and zero for dense data.

4 Check of the assumptions

For the whole workflow several assumptions about the data or the results of certain transformations are made. These assumptions have to be met at least roughly to ensure the correctness of the results.
This section gives checks for most of the assumptions like accuracy of the PSF scale estimation or that the matched filter of the matching size compared to the PSF performs best.
Also a discussion about the design choices is present in this chapter.

4.1 Calibration measurement

As described by Newberry (1998) the gain can be determined using the mean and variance of data with different mean intensities. Therefore a calibration dataset was acquired. The temperature and the gain factor were set to match the settings usually used. A sample with cells was prepared for the Storm measurement. First, eleven time series were taken, with open shutter and with increasing exposure time from 0 milliseconds up to 1000 milliseconds. Afterwards the same procedure was repeated with the shutter closed. Figure 4.1 shows the results for the first 6 exposure times. All points lie on a straight line. The gain factor determined by the calibration measurement was 3.9, the offset 315. This offset is too small which was found out by comparing the offset to the intensities from closed shutter measurements. An other method was used to determine the offset. There were also an other series of images taken with closed shutter. This series shows the response of the camera without any light around. The mean value of the shortest exposure time was taken as offset. Its value is 380.

4.2 Correction to Poisson distributions

It is very important for the whole workflow that the first transform results in background intensities that follow a Poisson distribution. To test this, a real world image was transformed with parameters taken from the calibration measurement described above. The offset was estimated from the minimal intensity of the raw image. The histograms for two generic background pixels are shown in Figure 4.2. A Poisson distribution with the mean of the pixels intensities is also shown in the figures. These histograms were acquired using the pixel intensities of one pixel for 3000 frames. The histogram matches the expected Poisson distribution.

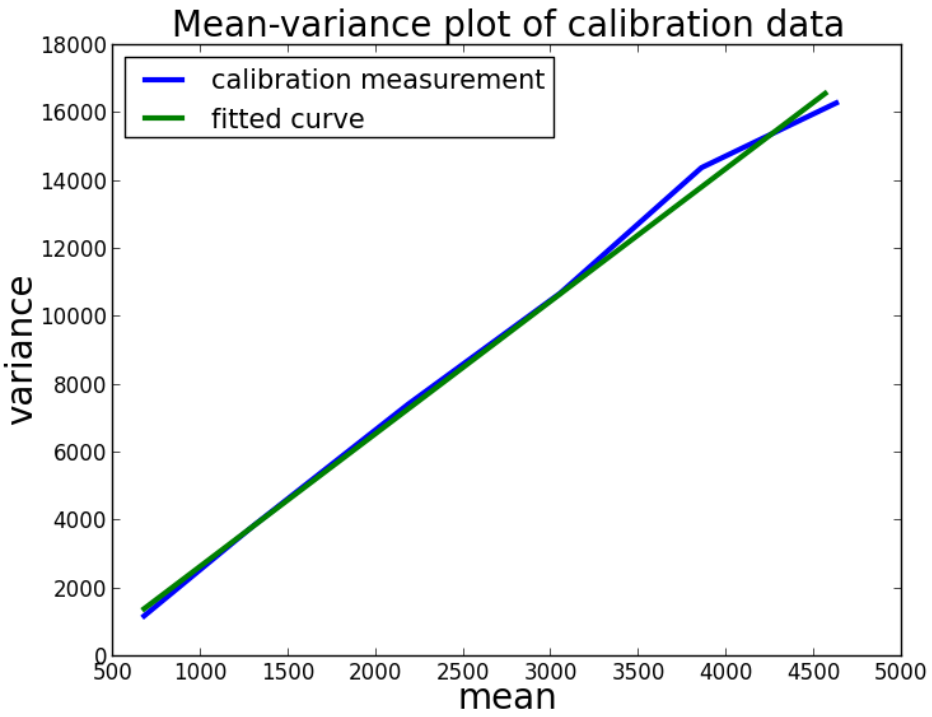
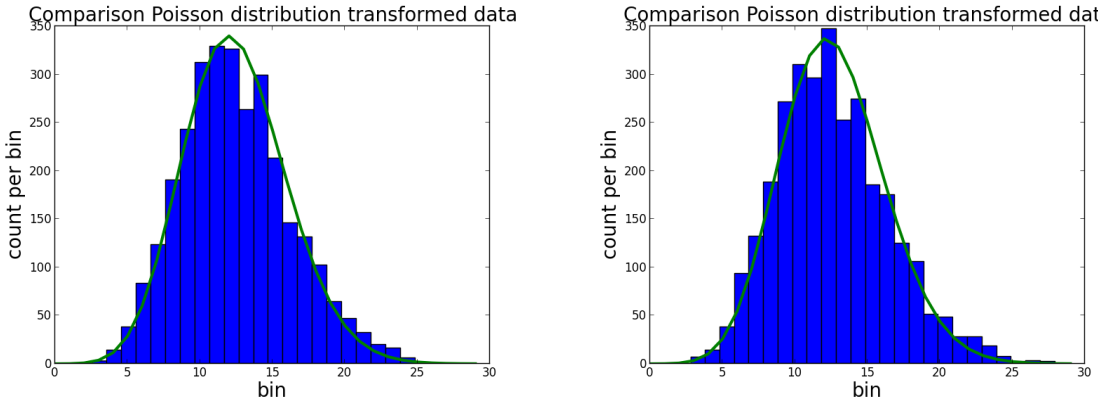


Figure 4.1: Result of the calibration measurements. The gain determined from this variance-mean plot is 3.9.



(a) Tubulin2 frame 10

(b) Tubulin2 frame 1010

Figure 4.2: These pictures show how well the histograms of background pixels taken over the first 3000 frames, follow a Poisson distribution. For the gain the parameter from the calibration measurement of $g = 3.9$ was used. The offset was estimated from the minimal intensities of the original image.

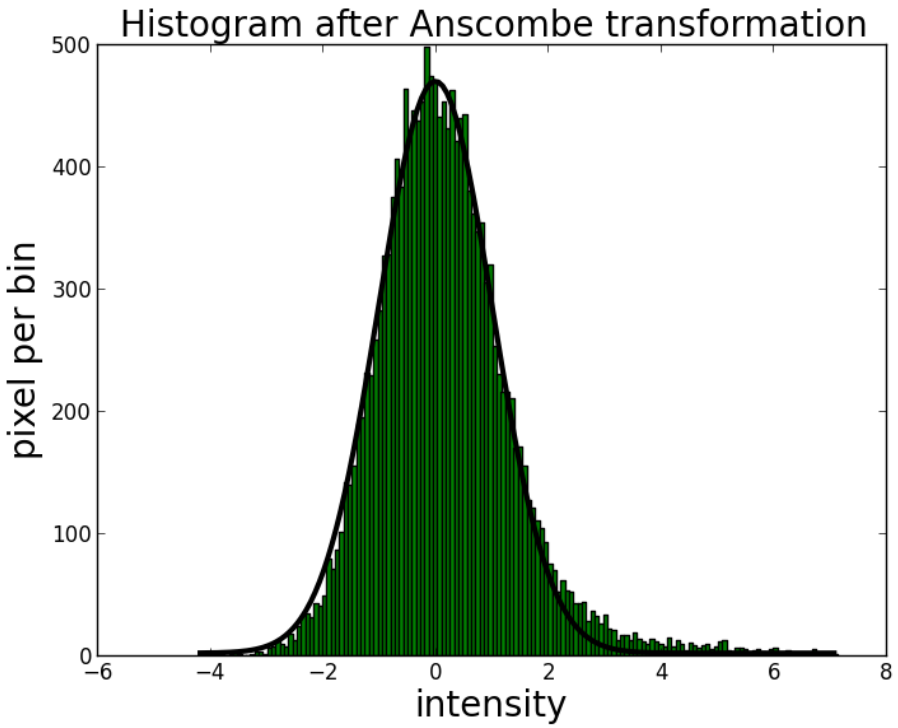


Figure 4.3: After the Anscombe transformation the background pixel intensities should be distributed with mean zero and variance one. This figure shows the histogram of the pixel intensities and a fitted Gaussian with variance one and mean zero.

4.3 Result Anscombe transformation

After the transformation described in section 2.4.1, the Anscombe transformation was applied to the same data set that was used in the previous section and the background was subtracted. Figure 4.3 shows the histogram of the intensities of one randomly chosen frame and a Gaussian with zero mean and unit variance. The histogram fits very well to the Gaussian. The tail exceeding the Gaussian on the right side can be explained by the signal that is present in the image and does not follow the Gaussian distribution, but has higher intensities.

There may be confusion why the Anscombe transformation that is applied to all intensities of one time step instead of all intensities of one pixel, does work. This is because the Anscombe transformation is applied to each intensity independently. Therefore all intensities at a certain time step are one sample from the Gaussian distribution in t dimension that results from the Anscombe transformation. All of these Gaussian distributions in t dimension have the same mean (since the background was subtracted) and variance (unit variance since Anscombe transformation). This means that the distribution created from all intensities of a certain frame has the same mean and variance as the transformed Poisson distributions in t dimension.

4.4 Accuracy of detection

Unfortunately the position of the fluorescent molecules can not be detected perfectly. There are two main contributions to the error in detection.
On one hand, there is a problem with finding the maximum in a noisy signal. Due to noise the spots maxima might be shifted.
On the other hand, the position of the fluorescent molecule is detected by upscaling the pixel grid and interpolation. After that the maximums position of the upscaled grid is taken as the resulting position. This gives an error from roughly a 12th pixel width. This error becomes less important with higher upscaling factors.

Because there is no groundtruth for the real data, test data and groundtruth had to be produce. This was done similar to the method described by Grüll et al. (2011). The difference was that, instead of generating just one spot per frame a different number of spots were simulated for each frame. For different signal-to -noise ratios, a dataset with 40 times 40 pixels and 1000 frames was created. Each frame containing one, three or five point spread functions. The same spots were used for each signal to noise ratio.

To determine the accuracy, the standard deviation between the true position of the signal and its detection were used. For data sets with more than one spot per frame, the best match within a certain distance around the true position was found. If a pair of true spot and detection were found, both were removed for further matching of the remaining signals. In the end the averaged standard deviation of the detection relative to the true positions were calculated as follows:

$$\text{std. dev.} = \sum_{i}^{\#\text{ pairs}} \sqrt{\left(\mathbf{p}^{\mathbf{i}}_{\text{true}} - \mathbf{p}^{\mathbf{i}}_{\text{detec}}\right)^2} \tag{4.1}$$

i runs over all found pairs of groundtruth and detections, $\mathbf{p}$ describes the two dimensional spatial vector of the groundtruth or detection respectively. False detections with no corresponding spot in the groundtruth did not contribute to the localization error.

The results can be seen in Figure 4.4. The Figure shows that the more spots are present per frame the harder it is to detect them properly. This is a result of the fact that spots that lay near each other might be detected as one spot. This gives rise to higher errors in the detection accuracy.

4.5 Matched filter is best filter

For denoising the image a matched filter is used. Since it is assumed that the point spread function is Gaussian, a two dimensional Gaussian filter with the estimated size of the point spread function is used. The following calculation shows why this is the best choice for the standard deviation.

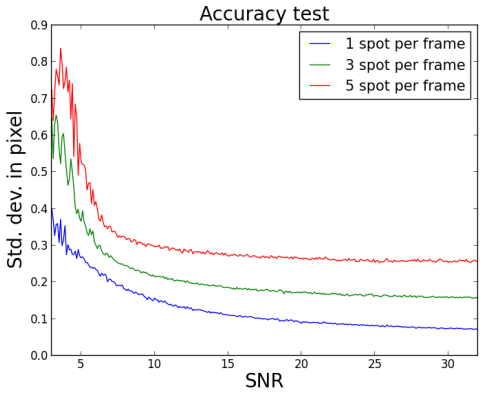


Figure 4.4: Result of the accuracy test. For datasets with one, three or five point spread functions per frame, evaluated for different signal to noise levels. The more dense the spots are the less accurate the detections are. The PSFs standard deviation is 1.4.

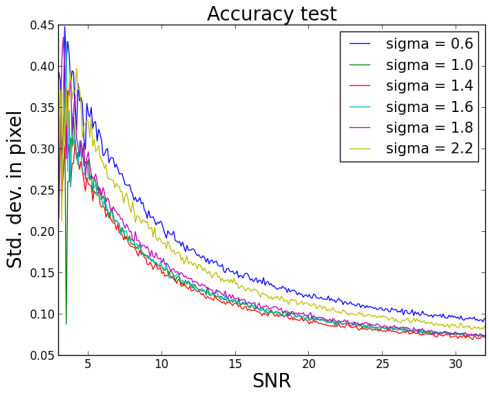


Figure 4.5: Result of the accuracy test for different sigmas for the Gaussian smoothing. One data set was processed with different sigma values and evaluated for different signal to noise levels. The true standard deviation of point spread function of the simulated data is 1.4.

This calculation was inspired by the calculation in Köthe (2007), page 207.
The assumptions are: the point spread function is Gaussian shaped with a true standard deviation of σ_{PSF}, the image contains white Gaussian noise with mean zero and unit variance, the applied filter for denoising is a Gaussian filter with standard deviation σ_{filter}.
The image f is composed of Gaussian filter convolved with the signal s and the white noise n mentioned above. s and n describe the filtered signal respectively noise.

$$f = s + n \tag{4.2}$$

Without noise the first derivative of f, which is equal to s in the noise-free case, vanishes at the center of the Gaussian signal. This center can be set to the origin without losing generality. But noise shifts the zero crossing of the first derivatives to $\Delta x,\ \Delta y$. For now just one dimension is considered.

$$\frac{\partial f}{\partial x}(\Delta x, \Delta y) = \frac{\partial s}{\partial x}(\Delta x, \Delta y) + \frac{\partial n}{\partial x}(\Delta x, \Delta y) \qquad = 0 \tag{4.3}$$

Using Taylor expansion around $x = 0$ at $y = 0$ for the signal, yields:

$$f_x(\Delta x, \Delta y) \approx \underbrace{s_x(0,0)}_{=0} + s_{xx}(0,0) \cdot \Delta x + n_x(\Delta x, \Delta y) \qquad = 0 \tag{4.4}$$

$$\Rightarrow \mathrm{var}(s_{xx}(0,0) \cdot \Delta x) = \mathrm{var}(n_x(\Delta x, \Delta y) \tag{4.5}$$

$$\Rightarrow \mathrm{var}(\Delta x) = \frac{\mathrm{var}(n_x(\Delta x, \Delta y)}{s_{xx}^2(0,0)} \tag{4.6}$$

The result for the variance of the first derivative of white noise convolved with a Gaussian filter of width σ_{filter} can be found at Köthe (2007), page 204. It is:

$$\mathrm{var}(n_x) = \frac{N^2}{8\pi\sigma_{\mathrm{filter}}^4}, \text{ with } N\text{: noise std. dev} \tag{4.7}$$

The convolution of the Gaussian PSF with an other Gaussian results in Gaussian with combined scales.

$$\mathcal{N}(0, \sigma_{\mathrm{PSF}}) * \mathcal{N}(0, \sigma_{\mathrm{filter}}) = \mathcal{N}\left(0, \underbrace{\sqrt{\sigma_{\mathrm{PSF}}^2 + \sigma_{\mathrm{filter}}^2}}_{=\sigma_{\mathrm{comb}}}\right) \tag{4.8}$$

The value of the second derivative in x-direction of the convolved PSF at (0,0) is:

$$\frac{\partial^2 S_0 \cdot \mathcal{N}\left((\mu_x, \mu_y), \sigma_{\mathrm{comb}}\right)}{\partial x^2} = -\frac{S_0}{2\pi\sigma_{\mathrm{comb}}^4} \tag{4.9}$$

Combining equations 4.6, 4.7 and 4.9 yields:

$$\mathrm{var}\left(\Delta x\right) = \frac{\frac{N^2}{8\pi\sigma_{\mathrm{filter}}^4}}{\frac{S_0^2}{4\pi^2\sigma_{\mathrm{comb}}^8}} \tag{4.10}$$

$$= \frac{N^2 \cdot 4\pi^2\sigma_{\mathrm{comb}}^8}{S_0^2 \cdot 8\pi\sigma_{\mathrm{filter}}^4} \tag{4.11}$$

$$= \frac{N^2}{S_0^2} \frac{\pi\left(\sigma_{\mathrm{filter}}^2 + \sigma_{\mathrm{PSF}}^2\right)^4}{2\sigma_{\mathrm{filter}}^4} \tag{4.12}$$

$$= \frac{N^2\pi}{2S_0^2} \left(1 + \frac{\sigma_{\mathrm{PSF}}^2}{\sigma_{\mathrm{filter}}^2}\right)^2 \left(\sigma_{\mathrm{filter}}^2 + \sigma_{\mathrm{PSF}}^2\right)^2 \tag{4.13}$$

The same calculation can be done with respect to y. This gives the same variance, yielding a total variance that is square root of 2 times larger, because the variances

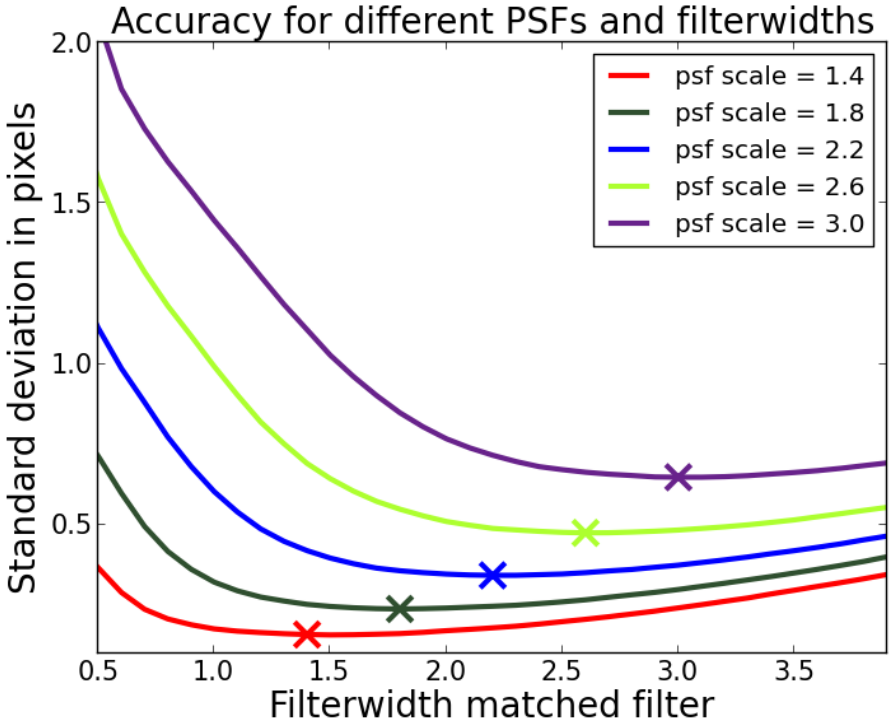


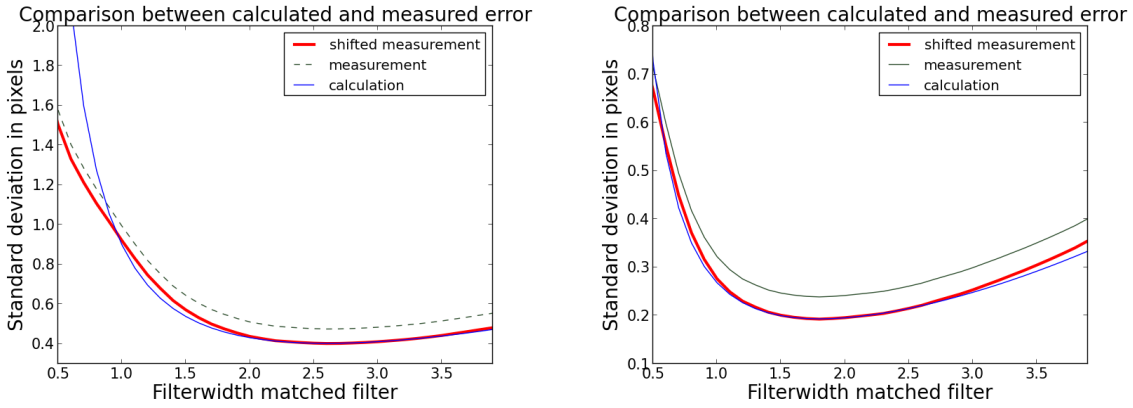
Figure 4.6: This figure shows the accuracy of detection achieved using different Gaussian filters for denoising. The cross marks the minimum of each curve.

are orthogonal. Therefore the standard deviation of the localization is:

$$\text{std. dev loc} = \frac{N}{S_0}\sqrt{\frac{\sqrt{2}\pi}{2}}\left(1+\frac{\sigma_{\text{PSF}}^2}{\sigma_{\text{filter}}^2}\right)\left(\sigma_{\text{filter}}^2+\sigma_{\text{PSF}}^2\right) \tag{4.14}$$

To verify these results test data sets have been created containing 3500 frames with one PSF per frame. Each created data set contained a PSF with different scale. This data sets have been filtered with Gaussian filters of different sizes. After that the maximum for each frame was detected and compared with the true position. Figure 4.6 shows the accuracy for different PSF scales and filter widths.
The best results were achieved using a Gaussian filter with the PSFs size. The larger the PSFs scale the more inaccurate the localization becomes.

The figures in 4.7 show the measurement, a shifted measurement curve and the result of the calculation. The measured error was larger than the calculated error, therefore the shifted line is also shown to state that the curves shape match but there is an additional error that shifts the line.
These results show that the calculated standard deviation of the localization can be used to estimate the localization error of SimpleSTORM for further calculations or error propagation.



(a) PSFs scale = 2.6 (b) PSFs scale = 1.8

Figure 4.7: Comparison between the measured and the calculated localization error. The dashed lines show the unchanged measurement results, for better comparison the line was shifted to match the calculated error.

It is important that the size of the ideal filter does not depend on whether an Anscombe transformation is applied to the data or not. The Anscombe transformation increases the standard deviation of the PSFs, this is shown in the next section. However the filter width that gives the lowes error is the original width of the PSF. This is shown in Figure 4.8. A data set with 3500 frames and a PSF with $\sigma = 1.5$ was created. Then the Anscombe transformation is applied to one copy of the data set. Both data sets were filtered with filters of different scale. The localization error is then plotted over the used filter scale.

4.6 Influence of Anscombe transformation on the PSF

The Anscombe transformation has no influence on the best matched filter, however the standard deviation of a PSF after applied Anscombe transformation changes compared to the original standard deviation. This change in the standard deviation depends on the SNR of the PSF. Figure 4.9 shows the change of the standard deviation of a PSF with $\sigma = 1$ for different SNRs. To avoid estimating a mean SNR for all PSFs used to calculate the mean power spectrum, an inverse Anscombe transformation is applied before the Fourier transformation is applied.
The reason why the inverse Anscombe transformation is used instead of not applying the Anscombe transformation, is that the data processing part is the same for all parts of the program (transformation to Poisson distributed background, background estimation and subtraction, Anscombe transformation). Therefore it is easier to do the inverse transformation instead of changing the data processing part.

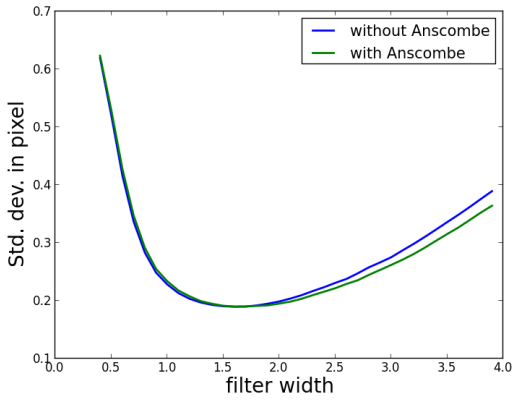


Figure 4.8: This plot shows the influence of the Anscombe transformation on the localization error for a PSF with $\sigma = 1.5$, which was filtered with Gaussian filters of different size. It can be seen, that the minimum of the localization error is the same for data with applied Anscombe transformation and for data without.

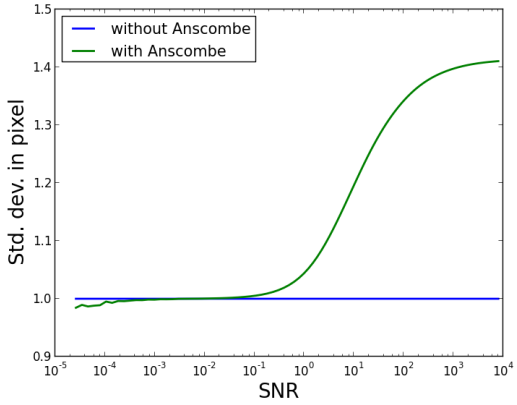


Figure 4.9: The standard deviation of the Anscombe transformed PSF changes with respect to the SNR.

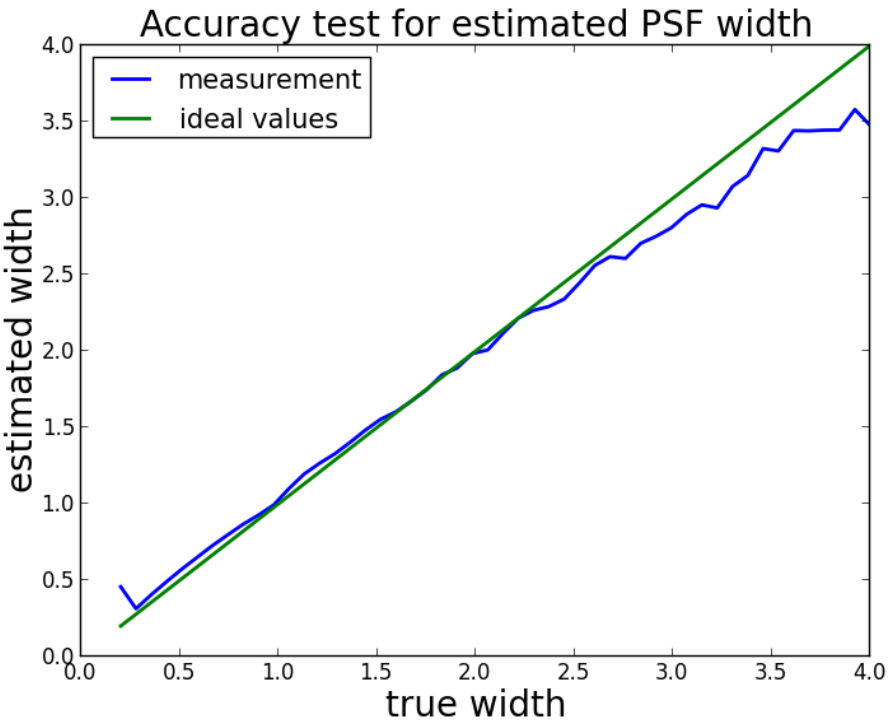


Figure 4.10: The figure shows the estimated widths of the point spread function plotted over the true width of the simulated signal. The signal-to-noise ratio was 10 for this test. The green line shows the simulated values.

4.7 Test PSF estimation

The estimation of the width of the point spread function of the signal is very important. The localization error increases with larger deviations of the matched filters width from the true width of the PSF, as shown in section 4.5.
Figure 4.10 shows the results of an accuracy test. For this test, data was created artificially. One PSF was simulated at a random location with a fixed width, Poisson noise was added. The signal-to-noise ratio was set to 10. Then this width was estimated by the SimpleSTORM algorithm. This was done with values for the standard deviation of the Gaussian in a range from 0.2 up to 4. The plot shows that for standard deviations up to 2.5 the estimated value for the width is very close to the actual value. The wider the PSF becomes in the spatial domain the smaller it gets in the Fourier domain. This is why the accuracy of estimation decreases for larger widths. This effect can be reduced by choosing a larger region of interest for the power spectrums accumulation. The parameter chosen for the region of interest for the plot in figure 4.10 was the default value. The typical width of the PSF for real world data is in a range of 1 to 2, in which the estimation gives reliable results.

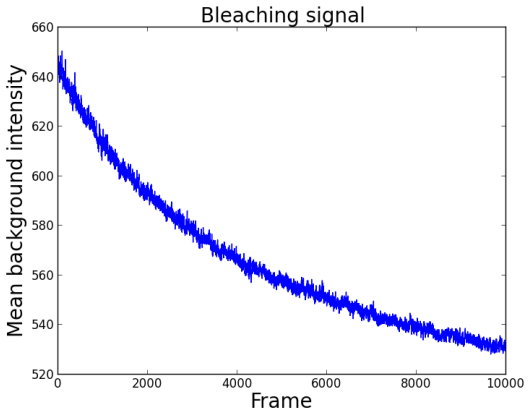


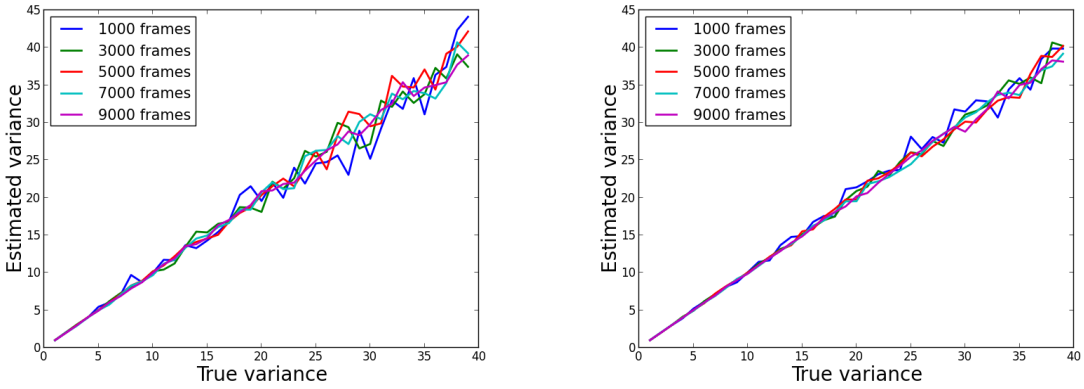
Figure 4.11: The number of active fluorophores decreases over time and so does the mean intensity of the image. This picture shows the mean intensities of 10 background pixel over time.

4.8 Bleaching signal

One assumption was that the backgrounds illumination is caused mainly from out of focus fluorophors and is therefore Poisson distributed. If this is the case the background illumination should decrease over time. Bleaching of fluorophores describes the process in which the fluorescent molecules change their conformation and lose the ability to emit light permanently. The spectrum of the emission can also change so that the fluorescent molecules can not be detected any more. All fluorophores used for the STORM images available bleach over time. The decay of the mean background intensity is shown in Figure 4.11.

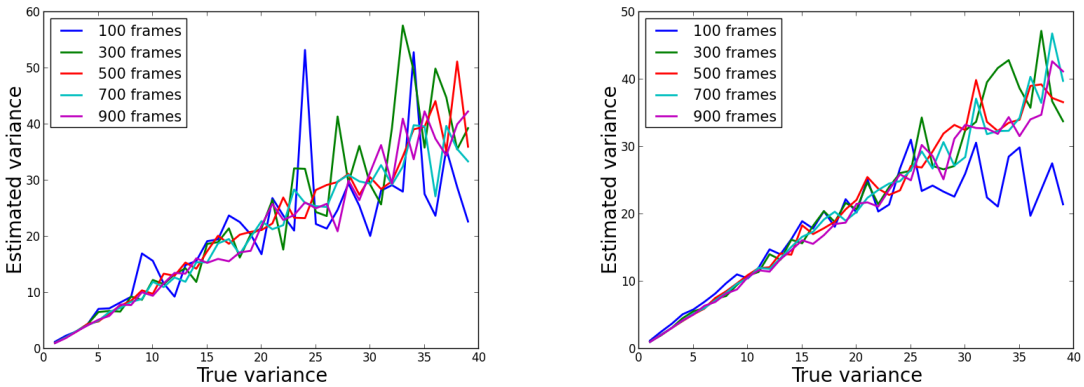
4.9 Reliability of skewness estimation

To estimate the true Poisson distributions mean using its skewness is a direct way to get to the true distributions mean regardless of the camera parameters.
This approach has one drawback. The larger the mean value of a Poisson distribution becomes, the more similar the Poisson distribution becomes to a Gaussian. This means that the skewness, which measures the asymmetry of the distribution gets smaller and smaller, converging to zero. The inverse of the skewness squared is used to estimate the mean of the Poisson distribution, this means that small uncertainties for the skewness lead to a high uncertainty for the estimated mean. Also a single distribution with a couple of hundred samples yields no reliable estimate. A robust estimate can be achieved if the median of the estimated mean of several distributions is calculated. Figure 4.12 show the results of the estimate of the mean using the skewness approach on patches of different size. The plot shows the estimated value on the ordinate, the true value on the abscissa. The different lines show the results for different sizes of the distribution. Figure 4.13 shows the same plot but for smaller distribution sizes. It can be seen, that the more pixels are used and the more samples



(a) 10 times 10 pixel patch (b) 20 times 20 pixel patch

Figure 4.12: Result of mean estimation using skewness approach.



(a) 10 times 10 pixel patch (b) 20 times 20 pixel patch

Figure 4.13: Result of mean estimation using skewness approach.

are available, the better the estimate becomes.

The estimate for the mean is accurate even for a smaller number of samples up to 15 to 20.

This approach has the drawback that for small data sets with only several hundred frames the accuracy of the estimated values is low. An other problem are artificially created data sets with a background intensity of 100 or above. In that case the skewness approach gives no reliable results.

4.10 Points lie on or above desired line

To estimate the camera gain and offset from the raw data the method described in section 2.5.1 is used. Ideally the intensities for every pixel would follow a Poisson distribution with a certain mean. However this assumption only holds for beads and

background pixel that never show any signal. The resulting points in the variance over mean plot will lie on a straight line. But what happens if background pixels are illuminated by a fluorophore in one or several frames?
This section proves that points in the variance over mean intensity plot lie on the straight line or above, but almost never below.
Set A contains n samples drawn from a Poisson distribution. The variance and mean of the set shall be μ_p. Set B is equal to set A but the last (n^{th}) member p_n^A is replaced by $p_n^A + c$. c is a positive value that describes the difference in intensity between a background pixel and an illuminated pixel. This models the additional intensity caused by a blinking fluorophore.

The exchange of the last pixel increased the mean intensity:

$$\mu_B = \frac{1}{n}\sum_{i=1}^{n}(p_i^B) \tag{4.15}$$

$$= \frac{1}{n}\left(\sum_{i=1}^{n-1}(p_i^A) + p_n^A + c\right) \tag{4.16}$$

$$= \frac{1}{n}\left(\sum_{i=1}^{n}(p_i^A) + c\right) \tag{4.17}$$

$$= \mu_p + \frac{c}{n} \tag{4.18}$$

The variance of set B times $(n-1)$ is given as:

$$(n-1)\cdot \mathrm{var}(B) = \sum_{i=1}^{n}\left(p_i^B - \left(\mu_p + \frac{c}{n}\right)\right)^2 \tag{4.19}$$

$$= \sum_{i=1}^{n}\left(\left(p_i^B\right)^2 - 2p_i^B\mu_p - \frac{2p_i^B c}{n} + \left(\mu_p + \frac{c}{n}\right)^2\right) \tag{4.20}$$

$$= \sum_{i=1}^{n}\left(\left(p_i^B\right)^2 - 2p_i^B\mu_p + \mu_p^2\right) + \sum_{i=1}^{n}\left(-\frac{2p_i^B c}{n} + \frac{2c\mu_p}{n} + \frac{c^2}{n^2}\right) \tag{4.21}$$

$$= \sum_{i=1}^{n-1}\Bigg(\underbrace{\left(p_i^A\right)^2}_{=p_i^B \text{for } i\neq n} - 2p_i^A\mu_p + \mu_p^2\Bigg) + \underbrace{\left(p_n^A + c\right)^2}_{=(p_n^B)^2} - 2\left(p_n^A + c\right)\mu_p + \mu_p^2 \tag{4.22}$$

$$+ \sum_{i=1}^{n}\left(-\frac{2p_i^B c}{n} + \frac{2c\mu_p}{n} + \frac{c^2}{n^2}\right)$$

$$= \sum_{i=1}^{n}\left(\left(p_i^A\right)^2 - 2p_i^A\mu_p + \mu_p^2\right) + 2p_n^A c + c^2 - 2c\mu_p \tag{4.23}$$

$$+ \sum_{i=1}^{n}\left(-\frac{2p_i^B c}{n} + \frac{2c\mu_p}{n} + \frac{c^2}{n^2}\right)$$

$$= (n-1)\mu_p + 2p_n^A c + c^2 + \frac{c^2}{n} - \sum_{i=1}^{n}\frac{2\left(p_i^B\right)c}{n} \tag{4.24}$$

$$= (n-1)\mu_p + 2p_n^A c + c^2 + \frac{c^2}{n} - \sum_{i=1}^{n-1}\frac{2\left(p_i^A\right)c}{n} - \frac{2\left(p_n^A + c\right)c}{n} \tag{4.25}$$

$$= (n-1)\mu_p + 2p_n^A c + c^2 + \frac{c^2}{n} - \sum_{i=1}^{n}\frac{2\left(p_i^A\right)c}{n} - \frac{2c^2}{n} \tag{4.26}$$

$$= (n-1)\mu_p + 2p_n^A c + c^2 + \frac{c^2}{n} - 2c\mu_p - \frac{2c^2}{n} \tag{4.27}$$

$$= (n-1)\mu_p + 2c\left(p_n^A - \mu_p\right) + c^2\left(1 - \frac{1}{n}\right) \tag{4.28}$$

The exchange of p_n^A increased the mean intensity, the variance of B must exceed $\mu_p + 1/n$ to lie above the line. Therefore the second part of the sum in equation 4.29

must be larger than $1/n$.

$$\mathrm{var}(B) = \mu_p + \underbrace{\frac{2c\left(p_n^A - \mu_p\right) + c^2\left(1 - \frac{1}{n}\right)}{n-1}}_{>\frac{1}{n}} \tag{4.29}$$

$$\frac{1}{n-1}\Big(2c\underbrace{\left(p_n^A - \mu_p\right)}_{>-\mu_p} + c^2\left(1 - \frac{1}{n}\right)\Big) \overset{!}{>} \frac{1}{n} \tag{4.30}$$

$$\Rightarrow c > \frac{\sqrt{\left(\mu_p^2+1\right)n^2 - 2n + 1} + \mu_p n}{n-1} \tag{4.31}$$

$$\frac{\sqrt{\mu_p^2 n^2 + n^2 - 2n + 1} + \mu_p n}{n-1} = \frac{\sqrt{\mu_p^2 n^2 + (n-1)^2} + \mu_p n}{n-1} \tag{4.32}$$

$$\leq \frac{2\mu_p n}{n-1} + 1, \text{because } \mu_p, n > 0 \text{ and } n > 2 \tag{4.33}$$

$$< 2\mu_p + 1 \tag{4.34}$$

This result confirms that if the additional intensity caused by signal c is at least two times the mean intensity plus one, the pixels variance rises.
If n samples from a Poisson distribution with mean μ_p are drawn, and one sample has the lowest value possible, for a Poisson distribution, one, at least two times the mean value has to be added to increase the variance. Lower values for c would decrease the summand $(p_n - \mu_p)^2$ in that specific case. The mean of the set increases if the replaced value lies further apart from the mean after replacement than before. This means that the probability to increase the mean by adding a positive value to one member of the distribution increases from 50 % for $c = 0$ to 100 % for $c > 2 * \mu$.
The variance increases even stronger when the pixel is illuminated by a fluorophore more than once. The variance is not affected by a constant offset, but increases even more rapidly if a gain factor larger than one is present. In case of a gain factor g the equations would get multiplied by g^2.

Figure 4.14 shows a scatter plot of simulated data. For this plot a normal data set with 200 frames was simulated. The data set contains background pixel with mean intensities between five and ten and beads with intensities between 80 and 160. For each pixel a Poisson distribution with the given mean is drawn. 30 PSFs per frame with a standard deviation of two and an intensity of 80 were added. The data was multiplied by 4 and an offset of 400 was added. Only a subset of all points is plotted. In Figure 4.14 the blue line shows the expected position for pixels with undisturbed intensity. Pixels that were not affected by any signal are displayed as green dots. The color of all the other pixel is calculated based on the value c added by the PSF. A red color is used if the added c is larger than the mean intensity, a blue color is used if the additional intensity is lower than the mean value of the pixel. The brighter the

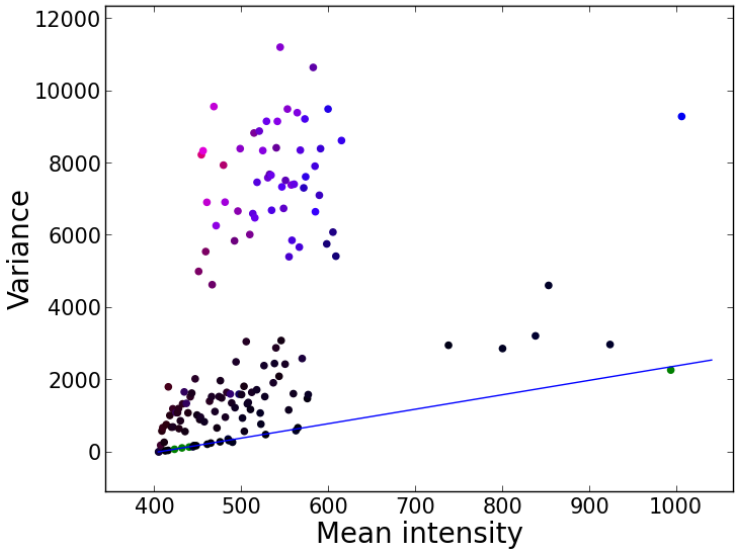


Figure 4.14: Variance mean scatter plot. The blue line indicates where pixels with constant mean would be displayed.

dots are the more often they are affected by signal. It can be seen, that the points tend to lie above the line, even when less than the mean value is added.

4.11 Variance vs Skellam

For the estimation of the camera gain and offset the method described in section 2.5 is used. There are two different ways to estimate the variance of pixel intensities.
The variance can be computed directly or the Skellam distribution as described in section 2.2.3 can be used.
The problem is to estimate the variance of the Poisson distribution of the pixels as good as possible, with as few samples as possible. The condition to work well on as few frames as possible arises, because there may be data sets with only 200 or 300 frames or the user does not want to spend more time on the parameter estimation.
To test which method for the variance estimation performs best, from a Poisson distribution with variance one, 200 samples are drawn, multiple times. Both methods were used to estimate the true variance. From all estimations the mean value of the estimated variance and the variance of the estimated variance were calculated. After that the procedure was repeated with increased true variance.
Figure 4.15 shows the results of for the mean of the estimated variances, Figure 4.16 shows the variance of the estimated variances, plotted over the true variance of the Poisson distribution.
Figure 4.15 shows, the mean estimates for the variance of a Poisson distribution with variances between zero and 40. The mean of all estimates from the Skellam approach is almost the same as the true variance (a straight line with slope one and

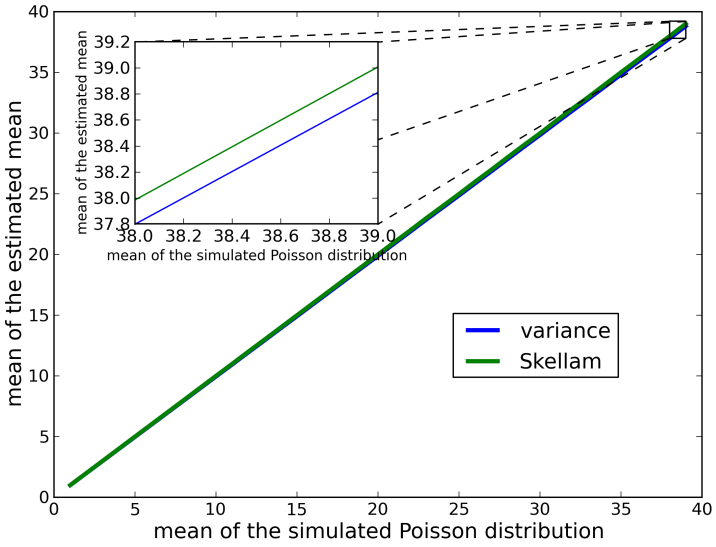


Figure 4.15: Mean of the estimated variances.

zero intercept). The variance approach underestimates the true variance. Figure 4.16 shows the variances of the estimates for both approaches. The variance approach has a lower variance than the Skellam approach.
Only one estimate is calculated in practice, therefore the variance approach is taken, because of the lower variance for the estimated values.

4.12 Best line fit method

To get the correct gain and offset based on the mean intensities and the corresponding variances of the pixels intensities, a robust fitting method is needed. This is because of the many outlier as seen in Figure 4.14. In this section four different methods will be described and their performances compared.

4.12.1 Different methods

Based on minimum covariance determinant (Fit1)

The base of this fitting approach is to find a compact subset of points that represents the inliers. To do so a R function based on the algorithm of Rousseeuw and Driessen (1999) is used. For this algorithm many initial subsets of the data are drawn. For each subset of points the minimal covariance determinant is calculated. The subset with the smallest covariance determinant is taken. Weighted linear regression is performed on the selected subset of points. Therefore the x range is split into several bins. For each bin the variance of the points is calculated and its inverse is used as weights for linear regression.
This fitting algorithm was developed and implemented in cooperation with Ilia Kats.

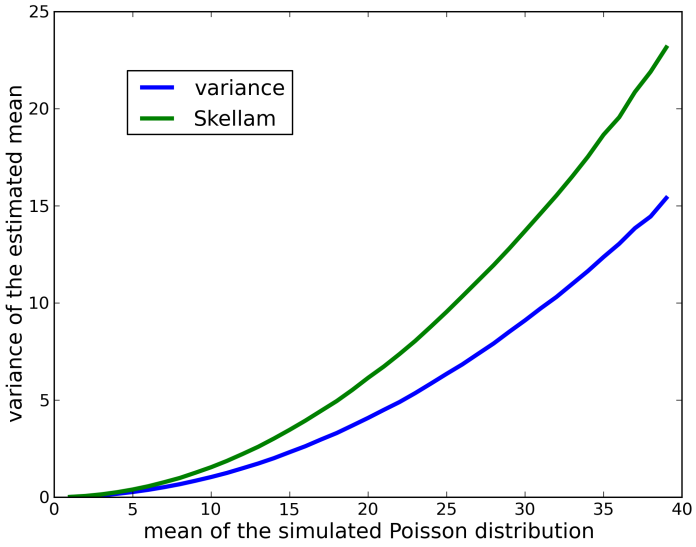


Figure 4.16: Variance of the estimated variances.

Use just lowest variances, weighted (Fit2)

Thousand iterations are used. For each iteration half of the preselected points is chosen randomly. The mean values are split into bins of equal width. For each bin the variance of the variances is calculated. Within each bin the point with the lowest variance is found. Linear regression is performed on all this lowest points with the inverse variance of variances of each bin as weights.

Linear fit of a random number of points (Fit3)

The difference to the previous approach is that the lowest points within each bin are found based on all data, then a linear fit is performed on a subset of varying size several times.

RANSAC (Fit4) and weighted RANSAC (Fit5)

A standard RANSAC (random sample consensus) algorithm is used. Its pseudo code looks like this:

```
input:
    data - set of mean intensities and variances
    model - straight line
    k - number iterations
    t - threshold for determining if a point fits the model
    d - number of close points that must be found at least not to
        discard the found model instantly
output:
```

```
    slope m and intercept c of the straight line

iteration = 0
best_m = 0
best_c = 0
best_error = inf
while iterations < k
    initial_guess_set = 2 random points from data
    m,c = fit_line(initial_guess_set)
    for every point in data
        if |y_point - m*x_point+c| < t
            add point to consensus_set
    m,c,current_error = fit_line(consensus_set)
    if len(consensus_set)>d and current_error < best_error
        best_error = current_error
        best_m = m
        best_c = c

return best_m, best_c
```

For the weighted RANSAC instead of a linear fit, a weighted line fit is done. The weights are determined as for the other weighted fitting methods.

Simple fit (Fit6)

For this very simple approach, a certain number of points is chosen randomly from all points. A straight line is fitted and the mean square error for the current set of points is computed. This is done several times. The best fit with the lowest mean square error is taken as the final result.

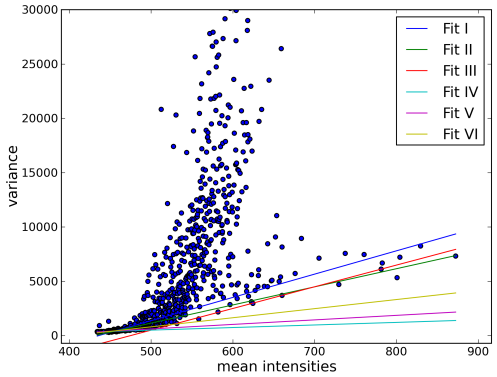
4.12.2 Discussion

The goal is to find a robust fitting method. Figures 4.17 and 4.18 show scatter plots of the means and variances of the preselected points and the results of the fits. It is difficult to determine the best fit by looking at the results, because there is not only one line but two or more visible. The calibration measurement yields a gain factor of 3.9 and a offset off 380. Table 4.1 shows the fit results for offset and gain. The results from both RANSAC methods (Fit 4/5) are closest to the results from the calibration measurement. This table also illustrates why for the offset the minimal value is used instead of the estimated value, there is much variation in the offset.
For the parameter estimation the weighted RANSAC method is used.
It is important to mention that the weighting that is used weights points from smaller intensities more than for higher intensities. The reason is that only the variance is considered which increases with increasing mean intensities. This is done on purpose. The lower intensities, originated by the background pixels show the desired line

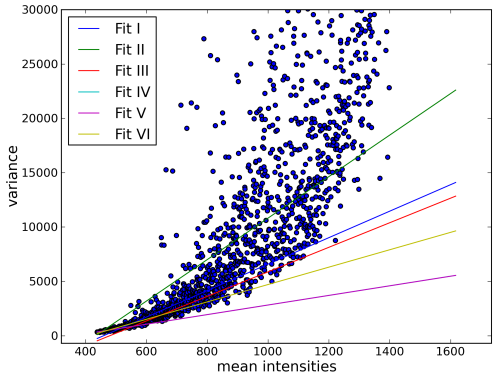
behaviour most clearly. An other advantage is that this approach is not absolutly dependend on beads, which give points in the scatter plot for high means.

Table 4.1: Fitting results for gain and offset for various data sets from our collaborators from Bioquant.

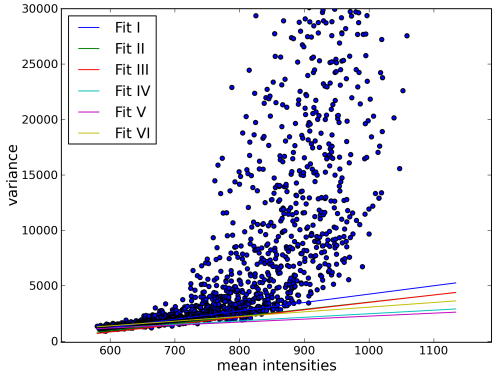
Data set	Fit 1		Fit 2		Fit 3		Fit 4		Fit 5		Fit 6	
	g	o	g	o	g	o	g	o	g	o	g	o
110215_HeLa1_Er647	21.4	435	16.6	430	20.0	474	2.37	282	4.17	350	8.37	402
110302_ER-Alexa647	12.3	458	19.1	432	11.3	479	4.42	358	4.42	358	8.00	410
110302_HeLa1_ER647	7.56	437	6.45	452	6.68	474	3.22	228	2.69	156	4.21	269
Pos2_2_green	12.6	484	7.05	452	7.12	483	3.13	308	3.44	329	5.76	407
Pos2_2_red	6.67	378	7.26	443	7.78	474	3.82	374	4.31	380	6.18	396
Pos11_2_green	12.9	442	8.17	395	9.15	474	4.51	376	4.59	378	5.83	390
Pos11_2_red	11.0	390	20.3	839	10.6	443	3.78	359	3.84	361	9.89	416
Tetra1_high_568	5.33	387	4.74	394	3.97	387	4.11	379	4.12	380	4.68	382



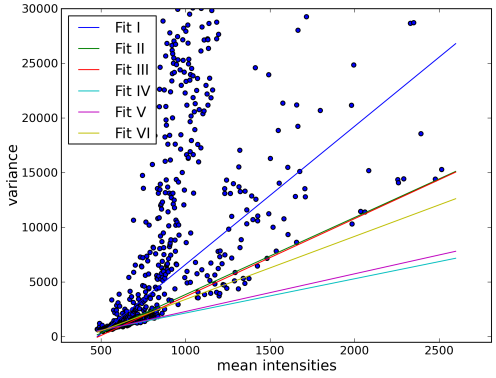
(a) 110215_HeLa1_Er647



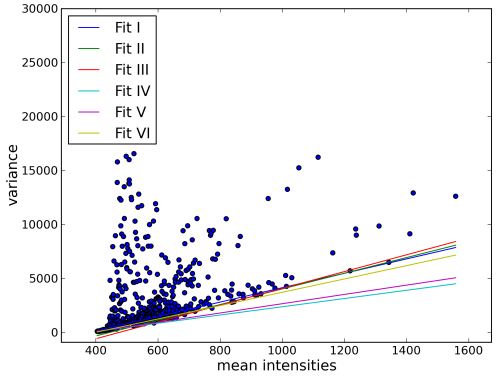
(b) 110302_ER-Alexa647



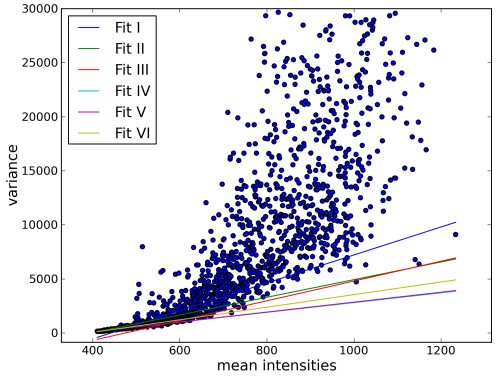
(c) 110302_HeLa1_ER647



(d) Pos2_2_green

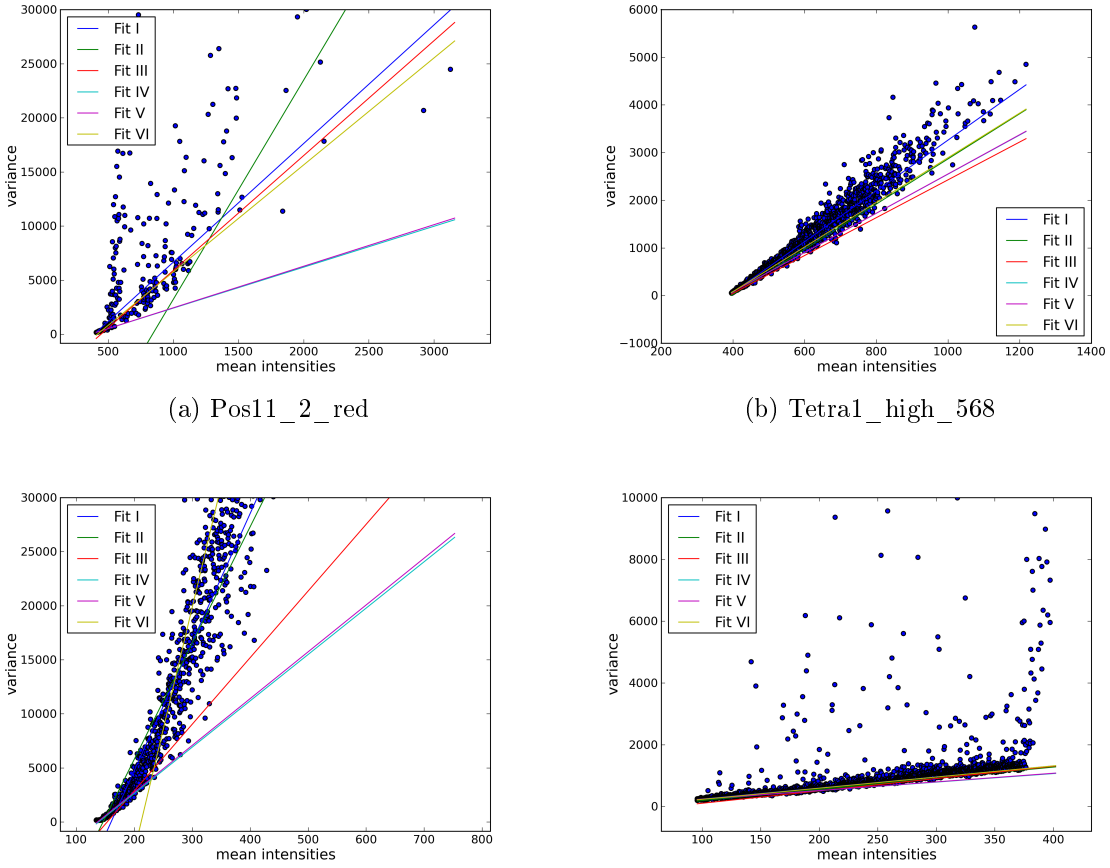


(e) Pos2_2_red



(f) Pos11_2_green

Figure 4.17: Scatter plot variance over mean for different data sets and the fitting results.



(a) Pos11_2_red

(b) Tetra1_high_568

(c) Tubulin1

(d) Bundled tubes high density

Figure 4.18: Scatter plot variance over mean for different data sets and the fitting results.

5 Multicolor registration

Biological samples are often labeled with more than one kind of fluorophore to show different structures within the sample. To align images from different color channels despite chromatic aberration, beads are used. Beads are fluorophores added to the probe that emit light in all wavelengths the different markers do, and therefore are visible in all channels. The beads can be used as landmarks, because their position in the original image must be the same. The task is to find a transformation that maps corresponding beads on each other.
The aligned images can then be investigated for colocalized structures.
Both tasks can be done using the improved version of the Colorcomposer.

5.1 Chromatic aberration

In microscopy it is often desirable to label different structures in a cell with different colors. To do so our collaborators use different fluoroscent molecules that emit light at different and therefore distinguishable wavelengths. Using appropriate filters it is possible to capture pictures containing light emitted from only one kind of fluorophore.
The propagation speed of light depends on the substance it is propagating through. The property of the substance that describes the ratio of the speed of light in vacuum c and in the substance v is called refraction index $n = c/v$.
The refraction index for one substance is wavelength dependent. Different wavelengths are refracted under different angles. This effect causes the splitting of white light into its components when shining through a prism.
The same effect causes the focal length of the microscope to be wavelength dependent. This means that the light for the same spot but with different wavelengths is not mapped to the same spot in the image. A sketch is shown in Figure 5.1.

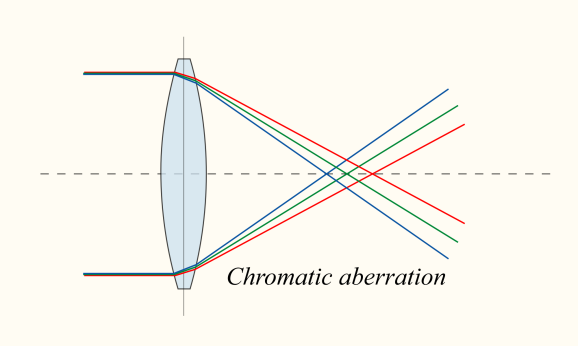


Figure 5.1: Sketch of the dependency of the focal length on the wavelength.

5.2 Colorcomposer GUI

The goal of the Colorcomposer tool is to provide software that is easy to use, flexible and powerful. The current version of the Colorcomposer is easy to use, because the different channels can be aligned by selecting the auto-align option.
Beads can also be manually selected or deleted, if the user wishes to do so. After the transformation the user can use the implemented tool for colocalization detection or save the transformed images and process them with the tool of his or her choice.
The Colorcomposer is powerful as it provides information about the number of points currently under the cursor or its intensity. The estimated transformation error and the localization error are computed and stored.
The basic framework for the Colorcomposer was set up by Schleicher (2011). It contained the workflow for importing and exporting images the handling of bead objects and a linear transformation that used the beads in the order they were found. This early version was unpractical.
Figure 5.2 shows improved Colorcomposer GUI with two datasets loaded. The buttons on the right give the user the option to add or remove beads in addition to the autodetected beads. There are different sliders to control the values used for bead detection. In the lower right corner additional information is provided about the total number of points within a rectangle with the selected cursor radius' size. The sum of the intensities and the total number of frames for each data set are also given. This information is helpful to determine whether a cluster of points is a bead or not.
At the top there is a menu bar with new options, that enable the user to discard all beads, automatically detect beads, calculate colocalization measures and show or hide the colocalization heatmap.

5.3 Features of the Colorcomposer application

5.3.1 Invariance of input data units

The resulting coordinate files from SimpleSTORM or other STORM algorithms may be given in units of pixels relatively to the unprocessed data or in nanometers. Treating coordinates given in nanometer as pixel units would lead to very huge and very sparse images displayed in the Colorcomposer. Therefore the Colorcomposer reads out additional information from the coordinates text files header. These information are the pixel to nanometer ratio and the used factor. With this information the picture can be reconstructed as an upsampled version of the input image by the used factor, regardless of the units used to save the coordinates file. If none of this information is given a pixel to nanometer ratio of 1 is assumed which guarantees backward compatibility with older coordinate files given in pixel units.

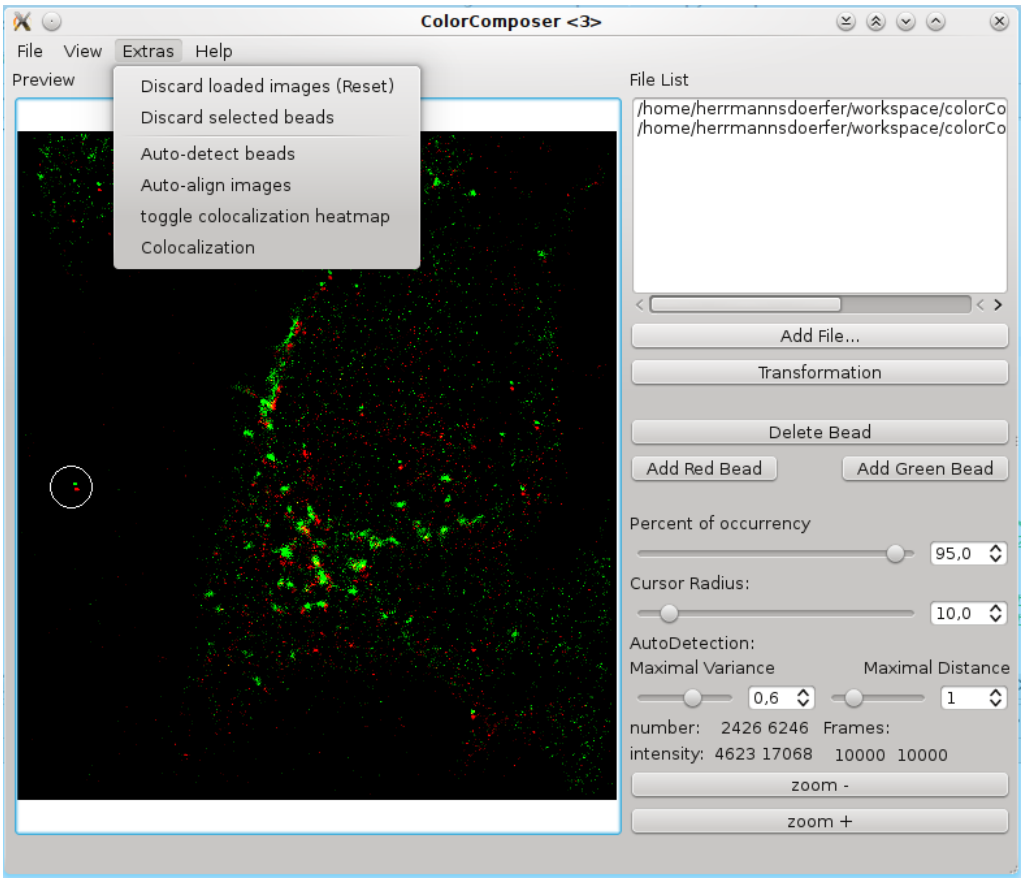


Figure 5.2: Improved Colorcomposer GUI. On the right are the sliders to set the parameters. There are also buttons to add or delete single beads. In the lower right additional information about the area under the cursor is displayed, such as numbers of pixels, total intensity and the number of frames in total.

5.3.2 Manual bead selection and removal

With the improved version of the Colorcomposer it is possible to add beads manually. To do so the desired location is clicked in the preview image and after that either the button "add green bead" or "add red bead" is hit. If there are enough frames containing localizations near the given location a bead is added to the center of mass of the intensities in that area. If the button "delete bead" is pressed all beads in the selected area are deleted.
This feature can be used to add beads missed by the automatic bead detection.

5.3.3 Automatic bead detection

The input for the Colorcomposer application is a text file created by the storm algorithm that contains information about the position, intensity, symmetry, frame number and signal-to-noise ratio of each detection. The beads should ideally appear in most of the images. This means they can be found by searching for detections that appear in almost every frame at the same position.
There was already an automatic bead detection implemented by Joachim Schleicher. This was improved as follows.
All important parameters for the bead detection can now be set in the GUI. The bead detection works by searching for points that appear in most of the frames. Instead of taking all localizations from the first frame as expected bead positions without considering locations that do not appear in the first frame, now a good subset of positions from the first 50 frames is found. This is done by skipping redundant positions based on the minimal distance of a new position to all positions already in the set. The range of 50 frames to look for beads is sufficient because it is very unlikely that all 50 detections of the bead have been missed.
After good candidates are found their number of points, variance and mean position is determined like described by Schleicher (2011).
In the end beads that are too close are merged to a new bead with its center right between the merged beads.

5.3.4 Alignment of two multicolor images

After the beads for each channel are found, the next task is to find the corresponding beads in each channel. It happens that some beads only occur in one channel. If this happens there will be no corresponding bead in the other channels.
The first step is to discard beads that are far away from any bead of the other color. From the resulting subset of beads the maximal number of possible bead pairs is estimated by taking the smaller number of beads per color.
The iteration begins that tries to find the best transformation to map all beads from one color to the beads of the other color. Therefore all beads from the color with less beads are chosen and the same number from the other color are selected. This selection is based on a probabilistic approach and the distance matrix containing the information about the distances between all beads of the two channels. It is more

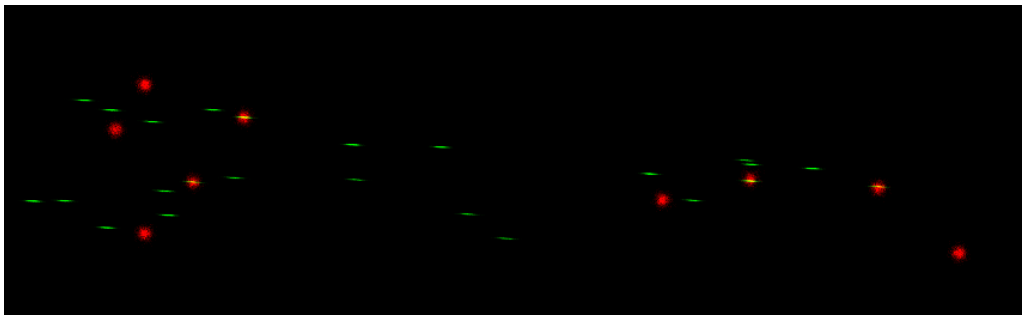


Figure 5.3: Affine transformation with shearing enabled. The green channel is deformed and one bead matches by chance which led to selection of this transformation.

likely for nearer beads of the other channel to be selected, but any bead within a certain range can be chosen.
Using these pairs of beads, a linear transformation is found, as described by Schleicher (2011).
This transformation is used to test how many other beads, that were not used to calculate the transformation match. It is assumed that the correct transformation will match additional bead pairs. This step is repeated multiple times.
The maximal number of matching bead pairs is compared with the number of assumed bead pairs. If less bead pairs are found than expected the number of expected bead pairs is reduced by one and the previous procedure is repeated with the appropriate number of beads randomly chosen from the channel with less beads.
All transformations that match more or equal bead pairs are stored. Also all bead pairs that are aligned under this transformation are stored. The number of assumed bead pairs is reduced by one and the iteration started again.
This loop ends when the assumed number of bead pairs is lower than 3, the minimal number of bead pairs necessary for the calculation of the transformation.
All candidates for the transformation are checked for the occurrence of shearing. In principle shearing should also be allowed for a linear transformation, but tests indicate that shearing does not occur, so it is disabled to improve stability.
If there are just three beads in each channel used for the transformation, then a perfect transformation is found every time, but the correct one, if possible, is the one without shearing. There will be an other problem with this transformation if the bead density is very high is possible that a transformation with much shearing is found. This compresses the beads of one channel to a slim band. The probability to find a matching point by chance is much greater then.
Figure 5.3 shows the result of an incorrect transformation of simulated data. The red channel was created by randomly placing beads. The green channel was slightly shifted and rotated. The green channel was transformed to match with the red channel. Just a subset of beads was used to calculate the transformation and so this solution was found and chosen from the algorithm because of the additional matching point.
This effects can be suppressed by not allowing shearing for the transformation.

5.3.5 Information about localization certainty

The detected spots from SimpleSTORM contain some localization error. This error will be derived in section 4.5. It depends on the signal-to-noise ratio and the scale of the point spread function of a single fluorophore.

Within all transformations that do not show shearing, the transformation is chosen that has the least root mean square error for all matching pairs of beads.

5.4 Total localization error

There are four contributions to the total localization error considered in this thesis. First there is the error that arises from the linear transformation to align the beads. To estimate this error the variance of the estimator is used. Since the transformation matrix is calculated by linear regression from the matrices B and R of the localizations of the first and the second layer of beads, with B being the matrix of the first layer to that the second layer R is transformed to. The variance of the estimator is

$$\text{variance registration} = \sigma^2 x_0 \left(R^T R\right)^{-1} x_0^T, \text{ with } x_0 = (x_o, x_1, 1) \tag{5.1}$$

Using equation 5.1 the variance σ^2 for all pixel can be calculated.
The second contribution to the total localization error is the localization error of the SimpleSTORM algorithm. Its formula is

$$\text{variance localization} = \frac{N^2\pi}{2S_0^2}\left(1+\frac{\sigma_{\text{PSF}}^2}{\sigma_{\text{filter}}^2}\right)^2\left(\sigma_{\text{filter}}^2+\sigma_{\text{PSF}}^2\right)^2 \tag{5.2}$$

It is derived in 4.5 and can be calculated for each detection individually based on the known signal-to-noise ratio. The signal-to-noise ratio gives directly the signals intensity S, because of the known noise variance of one which gives the nois' standard deviation N of also one. The PSFs width was either given or estimated and is passed to the Colorcomposer in the detection coordinate file's header along with the prefactor f for the actually used filter. Using this the localization error can be written as

$$\begin{aligned}\text{variance localization} &= \frac{N^2\pi}{2S_0^2}\left(1+\frac{\sigma_{\text{PSF}}^2}{f^2\sigma_{\text{PSF}}^2}\right)^2\left(f^2\sigma_{\text{PSF}}^2+\sigma_{\text{PSF}}^2\right)^2 &(5.3)\\ &= \frac{N^2\pi}{2S_0^2}\left(1+\frac{1}{f^2}\right)^2\left(1+f^2\right)^2\sigma_{\text{PSF}}^4 &(5.4)\\ &= \frac{N^2\pi}{2S_0^2}\left(2+\frac{1}{f^2}+f^2\right)^2\sigma_{\text{PSF}}^4 &(5.5)\end{aligned}$$

The third contribution results from the fact that the fluorophors are not directly attached to the sample of interest, since the antibodies used for staining are in between. The upper bound of this error can be estimated under the assumption that the target of the antibodies is much smaller than the antibodies, so that there is no occlusion in

the projection. This assumption does not hold in reality. Structures like cell membranes are certainly larger than the antibodies, but the assumption gives an upper bound.

$$\text{var x} = \frac{1}{\Omega_{\text{Sphere}}} \int\limits_{\Omega_{\text{Sphere}}} x^2 dV \tag{5.6}$$

$$= \frac{1}{\Omega_{\text{Sphere}}} \int\limits_0^{2\pi} \int\limits_0^{\pi} R^2 \sin\theta \left(R\cos\phi\sin\theta\right)^2 d\theta d\phi \tag{5.7}$$

$$= \frac{R^2}{4\pi} \int\limits_0^{2\pi} \int\limits_0^{\pi} \cos^2\phi \sin^3\theta d\theta d\phi \tag{5.8}$$

$$= \frac{R^2}{4\pi} \left[\frac{1}{2}\phi + \frac{1}{4}\sin 2\phi\right]_0^{2\pi} \int\limits_0^{\pi} \sin^3\theta d\theta \tag{5.9}$$

$$= \frac{R^2}{2} \left[\frac{1}{12}\cos 3\theta - \frac{3}{4}\cos\theta\right]_0^{\pi} \tag{5.10}$$

$$= \frac{2}{3}R^2 \tag{5.11}$$

R is the assumed distance of the fluorophore to the structure it is attached to (about 20 nm/0.09 px).
The fourth and minor contribution to the total localization error for each pixel is the quantization noise. It occurs when continuous intensities, as the photons forming the PSF, are transformed into integer values. Quantization noise describes the round-off error. The round-off error can be any value between -0.5 and 0.5 and is uniformly distributed. Its variance σ_Q^2 is

$$\sigma_Q^2 = \int\limits_{-\frac{1}{2}}^{\frac{1}{2}} x^2 dx = \frac{1}{12} \tag{5.12}$$

The unit of this error is the pixel size of the upscaled image and therefore more and more negligible the higher the upscaling factor is.
To get the total localization error all four variances are summed up for each pixel.

5.5 Colocalization

Colocalization in wide field microscopy is a measure of the overlap of data point from different channels. It can provide information whether or not two molecules interact. With increasing resolution of the images colocalization becomes more and more a measure of similar structures near each other. Two structures cannot be at the very

same position in the cell. The Colorcomposer software provides both global and local colocalization measurements.

5.5.1 Global colocalization

The most common colocalization measure is Pearsons correlation coefficient (Rodgers and Nicewander (Feb., 1988)). It is given as:

$$\text{Pearson correlation coefficient} = \frac{\sum_{i=1}^{n}(X_i - \bar{X})(Y_i - \bar{Y})}{\sqrt{\sum_{i=1}^{n}(X_i - \bar{X})^2}\sqrt{\sum_{i=1}^{n}(Y_i - \bar{Y})^2}} \tag{5.13}$$

It is the ratio between the covariance between the points of two channels and their standard deviation.

Also the Manders correlation coefficients M_1 and M_2 and the overlap coefficient (E. M. M. Manders (1993)) are calculated. Two channels, a red and a green one, are assumed.

$$M_1 = \frac{\sum_i R_{i,\text{coloc}}}{\sum_i R_i} \qquad M_2 = \frac{\sum_i G_{i,\text{coloc}}}{\sum_i G_i} \tag{5.14}$$

With $R_{i,\text{coloc}} = R_i$ if $G_i > 0$ and $R_{i,\text{coloc}} = 0$ otherwise and $G_{i,\text{coloc}} = G_i$ if $R_i > 0$ and $G_{i,\text{coloc}} = 0$. G_i, R_i are the intensities of the pixel of the green and red channel. Only pixels with at least a small component of the other color are used.

$$\text{overlap coefficient} = \frac{\sum_i R_i \cdot G_i}{\sqrt{\sum_i (R_i)^2 \cdot \sum_i (G_i)^2}} \tag{5.15}$$

5.5.2 Local colocalization

Global colocalization has the drawback that there is just one value for the whole image, not yielding any information about where colocalization is localized. If there are regions in the image that show a lot of colocalization and other regions without colocalization the same colocalization coefficient might be achieved as if one channel is distributed randomly.
For local colocalization analysis the algorithm from Malkusch et al. (2011) is used. It was further developed to gain a speed boost and runs approximately 40 times faster now. This was achieved by using scipys (Jones et al. (2001–)) ckdtree function, which is a k-d tree implemented in C.
The concept of the localization algorithm is to compute a colocalization value for each coordinate. This is done by calculating the distribution of all detections of both channels around each detection within a maximal radius R. This maximal radius determines the longest distance for which colocalization can be detected.

Figures 5.5 and 5.6 show the results from the colocalization calculation for both the

red and green channel from Figure 5.4. The difference between Figure 5.5 and Figure 5.6 is that the maximal distance is larger for the later. This example shows that the larger the maximal distance the more colocalization is detected. The choice for the maximal distance depends strongly on the sample.

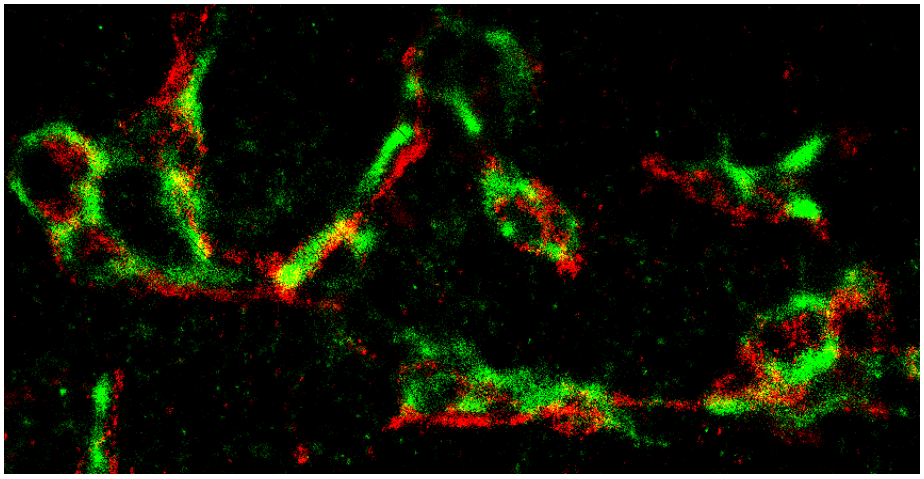
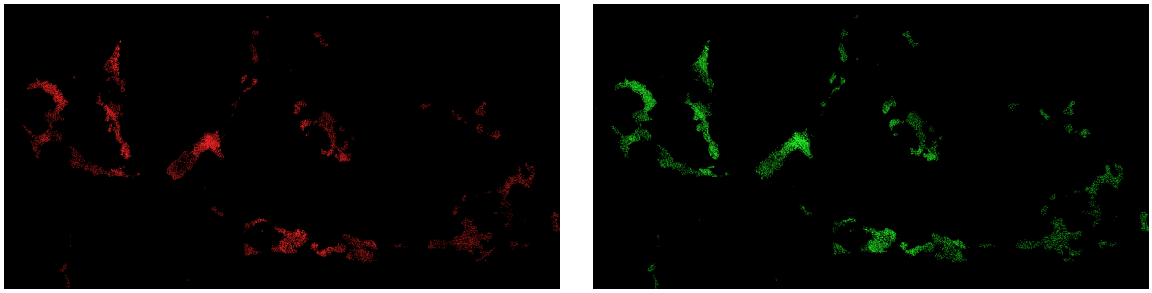
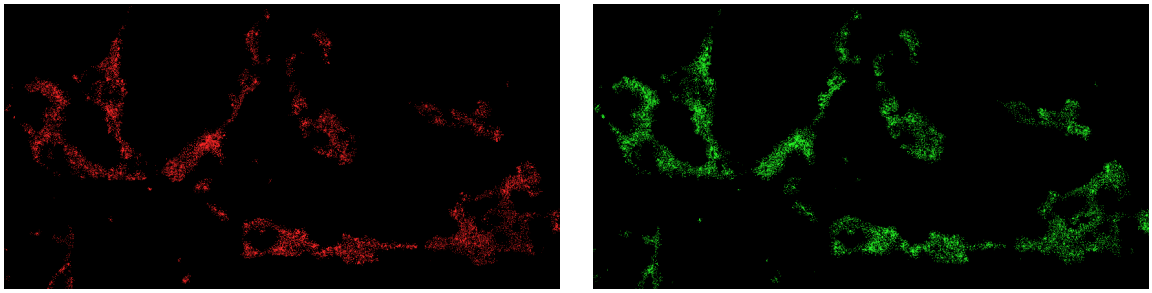


Figure 5.4: Aligned image showing colocalization.



(a) (b)

Figure 5.5: Colocalization for the red and green channel with a maximal radius $R = 50$ nm



(a) (b)

Figure 5.6: Colocalization for the red and green channel with a maximal radius $R = 100$ nm

6 Related work

There are many groups all over the world that have developed their own software to work on localization microscopy. This chapter shows the related work to give a better understanding what other concepts are used.
There are several algorithms that estimate the PSF from the data:

The DAOSTORM algorithm (Seamus J Holden (2011)) adapts algorithms from a software that is used to investigate crowded stellar fields. For spot localization a fit of multiple PSFs is used. This is done to small clusters of molecules and not globally. A model PSF is automatically generated from single PSFs in the data. DAOSTORM runs on the CPU.
FPGA Estimator (Grüll et al. (2011)) is an algorithm developed from a group at the Kirchoff-Institut für Physik in Heidelberg. It provides background subtraction by smoothing the pixels intensity over time, assuming Poisson distributed intensities. For high density data a method is used that sets all further pixel to zero once a local minimum is detected. A Gaussian estimator is used to determine the parameters of the Gaussian. The estimated scale can be compared with the given scale.
QuickPALM (Wolter et al. (2011)) uses the methods from rapidSTORM (WOLTER and SAUER (2012)) which was published earlier. RapidSTORM uses a Spalttiefpass filter and the Högborn "CLEAN" method for background suppression and a to select the local maxima, a Gaussian function with scales either given or estimated from the data is fitted. An amplitude threshold is used to distinguish between signal and background pixels. A maximum likelihood estimate of the PSFs parameters is used by GPUgaussMLE (Carlas S Smith and Lidke (2010)). For background subtraction either a constant threshold or a dark imaged acquired from all frames is used. Two filters of different size are applied to the data to determine signal candidates. A patch around the candidates is then used for maximum likelihood estimation of the PSFs parameters. Based on the estimated parameters filters are applied that compare these values with given input values for the PSF width. Also the uncertainty of the estimation was used to filter the candidates. This algorithm runs on the GPU and can also be applied to 3d data.

Most of the algorithms use a maximum likelihood estimator to localize the maxima. An other example that uses the same method as simpleSTORM is called M2LE (Starr et al. (2012)) which uses an user defined threshold for candidate selection and the median of a ROI for background suppression, an user specified ellipticity threshold which can be made intensity depended to discard candidates with an too high ellipticity. The PSFs position is determined using a maximum likelihood estimator, separated Gaussians are used for speed up.

There are algorithms that provide their functionality as plugins for ImageJ or python packages. This makes it easier to implement this methods in ones workflow.
There is the ImageJ plugin PeakFit (Alex D. Herbert (2013)) that uses a 2d Gaussian non-linear least squares Levenberg-Marquardt algorithm for fitting. The candidates are acquired by subtracting two filters of different sizes. The PSFs width can either be specified or estimated from the data. The fitted spots are filtered based on their signal-to-noise ratio. For high density data multiple Gaussians are fitted to the data. As an ImageJ plugin it can deal with any kind of data processable via ImageJ.
PYME (Baddeley et al. (2011)) is a python package with functionalities that can be used for localization microscopy and other forms of widefield microscopy. Besides the STORM data analysis it also provides reconstruction and postprocessing algorithms. It is applicable to 2d and 3d data. For noise treatment a Gaussian-Poisson noise mixture model is used.

7 ISBI Challenge 2013

7.1 Introduction

The goal of the ISBI Challenge, as announced on their website (Biomedical Imaging Group (2013)), is to give an overview and understanding of available algorithms for single particle localization microscopy. The focus was on 2d localization, to give information about the depth of a localized spot was optional. To benchmark results one needs groundtruth. Therefore the organizers created synthetic datasets of biologically relevant structures such as tubulins. To match realistic conditions the data was transformed to contain different kinds of noise and background.
The participants were given training data sets and the corresponding groundtruth. One month before the deadline of the challenge the test sets were released. There were two different kinds of datasets in principle. One with very dense spots and shorter sequences, the other with longer sequences and fewer spots per frame.
All participants were asked to submit their results, including thee it took to run the algorithm and the hardware configuration of the used system.

7.2 Terminology

To be able to compare different algorithms there must be a way to determine the correctly detected spots. To do so for each estimated position of a fluorophor, the nearest correct position of the molecule in the groundtruth data was searched within a lateral tolerance disc. Once a match was found these two spots were taken out of consideration for the matching.
One important parameter for this evaluation is the radius of the lateral tolerance disk, because it has great influence on the number of detections considered to be true positives (TP).
Detections with no associated spot in the groundtruth are called false positives (FP), spots in the groundtruth with no matching detection are called false negatives (FN). This matching is done frame by frame, it is not possible to match a point from different frames even if the x and y coordinates match perfectly.
The precision (p) of a classification task is defined as the ration between the number of true positives and the sum of true positives and false positives:

$$\text{precision: } p = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{7.1}$$

It is a number between 0 at worst and 1 at best, showing how reliable the result is, how likely it is that a labeled sample really belongs to the predicted class. In this

context it means how certain a detected spot has its origin in a fluorophore attached to the investigated structure and its origin is not wrongly detected background noise. An other important value is the recall r that is defined as the ratio of true positives and the sum of true positives and false negatives:

$$\text{recall: } r = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{7.2}$$

The recall lies also in a range from 0 to 1 and gives an impression on how many relevant spots were found.

7.3 Measures

For the evaluation three different measures were used. The f-score index f, the Jaccard index J and the root-mean square distance RSME.

F-score

The f-score f is directly computed from precision and recall.

$$\text{f-score: } f \quad = \frac{2 \cdot p \cdot r}{p + r} \tag{7.3}$$

It ranges from zero for bad performance up to one for perfect results. The f-score is combination of precision and recall.

Jaccard index

Let A be the set of points of the groundtruth and B be the set of detected points. The Jaccard index J is then defined as:

$$\text{Jaccard: } J = \frac{|A \cap B|}{|A \cup B|} \tag{7.4}$$

The intersection is done frame by frame. This means two spots from the groundtruth and the detection set only match if they occur in the same frame.

RSME

The root-mean square distance gives an impression on how big the squared distance between a spot in the groundtruth and an associated detection was in average. It can be calculated as follows:

$$\text{RSME} = \frac{1}{|A \cap B|} \sum_{i=1}^{|A \cap B|} \left(p_a(x, y) - p_b(x, y)\right)^2 \tag{7.5}$$

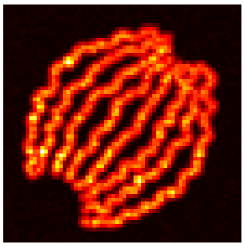
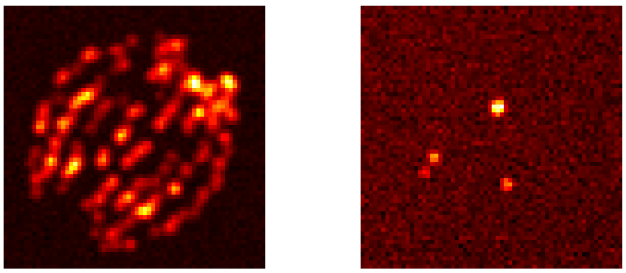


Figure 7.2: Maximum projection of bundled tubes data set



(a) High spot density (b) Low spot density

Figure 7.1: One frame from bundled tubes training data set

7.4 Training data

All pictures shown in this section are reconstructed from data as provided by the Biomedical Imaging Group (2013).

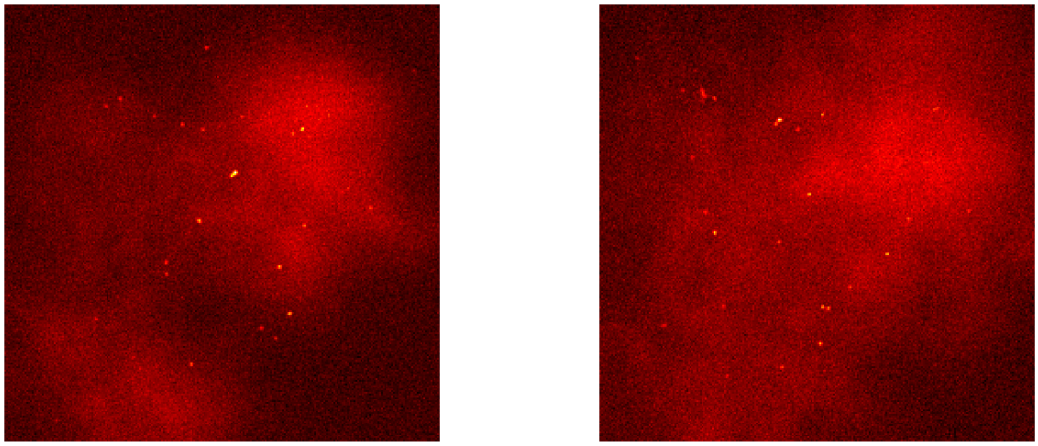
7.4.1 Bundled tubes datasets

There were two kinds of bundled tubes data sets, both created from the same underlying structure. One set with a high spot density and a short sequence of 360 frames, the other with fewer spots per frame but 12000 frames in total. Picture 7.1 shows one frame of each data set.

The original images were very small, with only 64 pixels in each dimension. Both sequences had spatial and temporal constant background. Figure 7.2 shows the maximum projection of the bundled tubes data set. The maximum projection is used to reduce the dimensionality of a data set. In this case for each pixel in x- and y-dimension the brightest value from all frames is taken.

7.4.2 Tubulin data sets

The other training data sets models 7 microtubules, a structure that is a long filament up to several micrometers long and with a diameter of about 25 nanometers. The spot density lies somewhere between the high density and the low density of the bundled



(a) Tubulin2 frame 10 (b) Tubulin2 frame 1010

Figure 7.3: This pictures show the variability of the background in the spatial and temporal dimensions

tubes data sets. These data sets show strong inhomogeneity in spatial dimensions and moderate inhomogeneity in temporal dimension, see Figure 7.3.

7.5 Submissions

Now the question arose what it means to be the best algorithm. Is the one the best that finds the most true positives regardless of the number of false positives and the accuracy of detection? Or is it more favorable to find exclusively correct spots, but less? There are different definitions for the best algorithm possible.
For that reason each dataset was processed with three different settings described below. All parameters were set manually.

7.5.1 High precision

A biological question to be answered by STORM microscopy can be: "How are the labeled proteins distributed over the cell?" There might be just a couple of fluorophors attached to each protein. In that case a high precision is crucial. Otherwise homogeneously distributed false positives all over the image might lead to the conclusion that the protein of interest is distributed all over the cell, instead of being clustered in some spots.
The high precision submission set the α value to low values to get almost 100 percent precision.

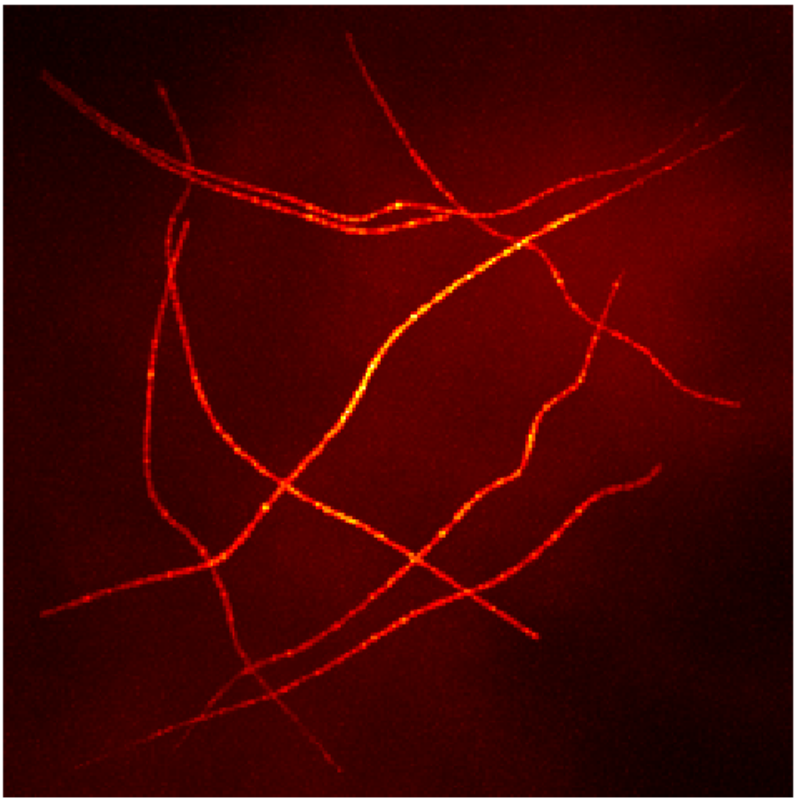
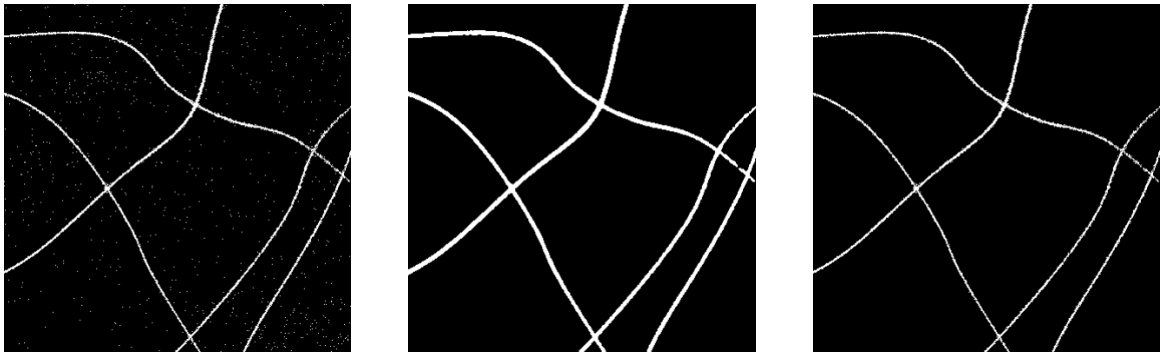


Figure 7.4: Maximum projection of tubulin data set



(a) Image before postprocessing (b) Mask (c) Result of postprocessing

Figure 7.5: Overview over the steps of postprocessing.

7.5.2 High score

The parameters for the high score submission were set to maximize the Jaccard index. This was done by increasing the α value and estimating the number of true positives and false positives that were found in addition compared to the previously used α value. Based on tests on the trainings date the highest Jaccard index is achieved if the number of new false positives is equal to the number of new true positives.

7.5.3 Highest score via postprocessing

The higher the α value the more likely it is to detect false positives. But these false positives are distributed all over the image. For the postprocessing the assumption is that the true structure in the image shows a higher density of spots than the background with false positives. A mask was created by smoothing the reconstructed image and then thresholding it. This results in a mask that covers image regions with high density. This mask was used to discard all detections which are not covered and are therefore assumed to be false positives in the background.
For the submission a high α was chosen to get also darker true positives. Afterwards the postprocessing was applied. The postprocessing worked well on the contest data because it contained only dense structures. For biological applications the postprocessing has to be used carefully because it discards small, isolated structures, which might be not noise but widely spread proteins.
Figure 7.5 shows the data before postprocessing, the mask and the result.

7.6 Results

All tables shown in this section and all diagrams are created from the results released by the Biomedical Imaging Group (2013).

7.6.1 High density data

Table 9.1, 9.2 and 9.3 show the results of the SimpleSTORM algorithm for the different submissions. The submission with postprocessing applied gave the best results for the Jaccard index, compared to the other two submissions. As expected the settings for high score, but without postprocessing yielded better Jaccard indices than the high precision settings, but also worse precision. As expected the high precision settings resulted in the highest possible precision of one hundred percent. Also the postprocessed data has a perfect precision. This demonstrates that the postprocessing removed the false positives, which decreased the precision in the high score (without postprocessing) results. The accuracy was the same for all three submissions.
The time the software needed was submitted with the results, this means that it depends strongly on the system the algorithms run on. Over all SimpleSTORM took about 90 seconds to process the high density data sets running on an ordinary laptop. This was an average runtime. Most of the faster algorithms used the GPU.

Figures 9.1 and 9.2 show the mean of the Jaccard index and the RSME score taken over all high density data sets. The results for SimpleSTORM are highlighted. SimpleSTORM achieved an average rank for the Jaccard index and a top five rank for the RSME score. With 100 percent precision it was among 6 other participants on the first rank.

7.6.2 Low density data

The tables 9.4, 9.5 and 9.6 show the same trends as for the high density data. But with this data sets, the difference between the precisions is greater between the high precision submission and the other two submissions. The runtime was about five minutes. This was, like for the high density data sets, average.

As for the high density data sets the mean scores were calculated for the low density data set and are shown in figure 9.3 and 9.2. The results for the precision are also shown in figure 9.5. SimpleSTORM achieved the third rank in the Jaccard index and intermediate ranks for accuracy and precision.

7.6.3 Overall

Of the 29 participants of the challenge just 15 submitted results for the high density data sets. There is no winner since no software performed well on each score. This can be explained by a trade-off between high accuracy (a low RSME) and the number of spots detected (influence on Jaccard index). For example, bright spots are easier to localize. Just finding the brightest spots yields a good performance on the RSME score but less points. This trade-off can be seen not just in our results, where a high Jaccard index goes along with an intermediate RSME score and vice versa, but also for other algorithms like Fast-ML-HD, which has the highest Jaccard index on the high density data but the second worst RSME score for the same data set.
To compare the different algorithms some metric has to be used that takes the shown trade-off between Jaccard index and accuracy into account. One way is to use the

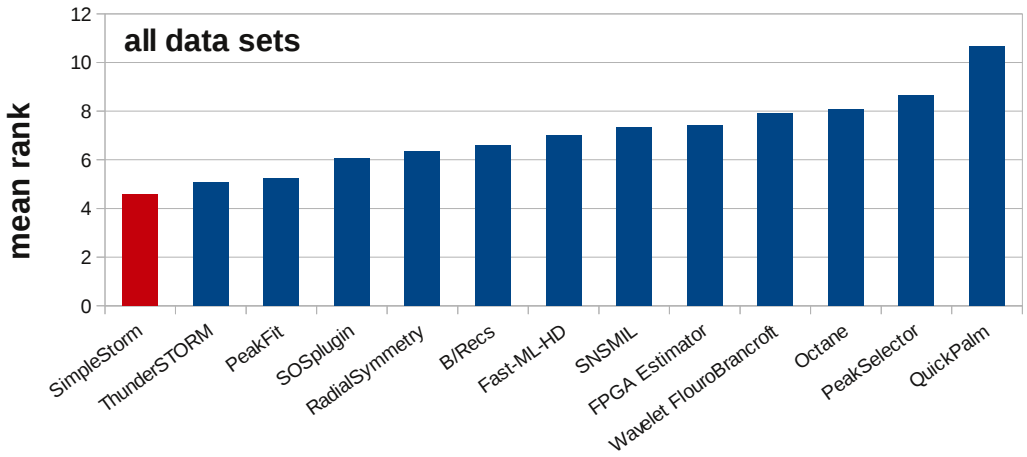


Figure 7.6: Averaged rank over Jaccard index and RSME score for all data sets. Lower ranks are better

rank each software achieved on the different data sets for different scores and average over it. The results averaging just the mean rank for the Jaccard index and for the RSME score are shown in figure 9.6 for the low density data sets and in figure 9.7 for the high density data sets.
SimpleSTORM achieves the best mean rank for high density data and the third best result for the low density data sets.
Figure 7.6 shows the result if the average rank for each data set and for RSME score and the Jaccard index is calculated based on the ranks of all algorithms that processed both high density and low density data only. All other algorithms were taken out of this consideration. Although this method of comparison did not weight the ranks which means that extraordinary good performance makes no difference to slightly better performance between two consecutive software, it shows that SimpleSTORM provides a good compromise between a high number of correct detections and high accuracy.

8 Summary and outlook

8.1 Summary

A new, easy to use and robust data processing algorithm was presented. It enables even users that are new to STORM imaging techniques to get reasonable results in a straight forward way. Understanding about several parameters and their influence on the algorithm is not necessary, as well as finding a good set of parameters.
SimpleSTORM is also a powerful tool that allows tweaking of the parameters. Its performance has been shown in the ISBI localization microscopy challenge where SimpleSTORM showed good performance over all data sets.
With the improved Colorcomposer software an easy to use and powerful tool for image reconstruction of multi-channel microscopy data is provided. The reconstructed images of different channels can be aligned automatically. A colocalization analysis can be performed.

8.2 Outlook

3d Storm

STORM imaging techniques have been extended to capture 3-D structures up to a thickness of micrometers with super resolution. This will become more and more important in the future. There are different ways to get the information about the depth. One way is to use a cylindrical lens which results in point spread functions that are symmetrical if the fluorophore lies in the confocal plane, but more and more asymmetric the further away the captured spot lies off the confocal plane.
Another way for depth estimation is to use normal lenses but use the information that the width of the PSF exceeds with greater distance of the spot to the confocal plane. Both methods might be implemented as only the part of the maximum detection has to be altered.

Improved method to detect maxima

Instead of looking for the maxima in an upsampled image, a model for the point spread functions could be used to find the maxima. This would make it possible to find maxima that are too close together so that their maxima merge into only one maxima between them.
This changes can be implemented without affecting the structure of the other parts of the SimpleSTORM software, just the maxima detection part must be changed.

Colorcomposer implemented in Storm-Gui

The Colorcomposer could be implemented in the SimpleSTORM-GUI. The benefits beside increased performance are an easier and more direct way of the processing and the need for only one tool.

Combination of Wiener filter and Gaussian filter

Depending on the spot density and the data attributes like the signal-to-noise ratio it might be favorable to combine the usage of a Wiener filter and a Gaussian filter to chose the filter that performs best on the data given.

Interactive parameter selection

An unprocessed image is shown to the user. The user marks areas with spots of interest, specially the darkest spots that should be detected. Then the program sets the parameters in a way that at the user labeled spots will be detected in the end. This would be a good way to set the best alpha value or the signal-to-noise ratio limit.

Alternatively one could set the parameters and get a direct feedback which points would be selected in the shown frame. If the results are not satisfying the parameters can be changed instantly.

9 Appendix

9.1 List of Figures

9.2 List of Tables

9.3 Additional tables of ISBI challenge results

Table 9.1: Results for the main submission for the high density data set (with postprocessing)

Dataset	Jaccard (in %)	Precsision (in %)	Recall (in %)	RMSE (in nm)
HD1	40.44	100	41	40.13
HD2	30.84	100	31	63.18
HD3	12.55	100	13	61.8

Table 9.2: Results for the high precission submission for the high density data set

Dataset	Jaccard (in %)	Precsision (in %)	Recall (in %)	RMSE (in nm)
HD1	37.62	100	38	40.23
HD2	28.37	100	28	60.70
HD3	12.66	100	13	62.58

Table 9.3: Results for the high score submission for the high density data set (without postprocessing)

Dataset	Jaccard (in %)	Precsision (in %)	Recall (in %)	RMSE (in nm)
HD1	37.96	100	38	40.22
HD2	29.73	94	30	63.33
HD3	13.00	99	13	63.95

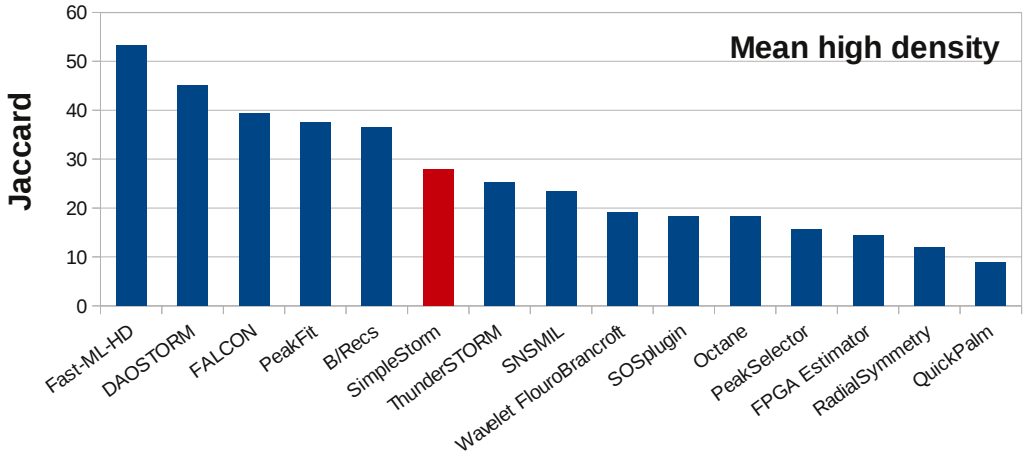


Figure 9.1: Results for high density data sets. The Jaccard indices are averaged over all three data sets. For this score higher is better.

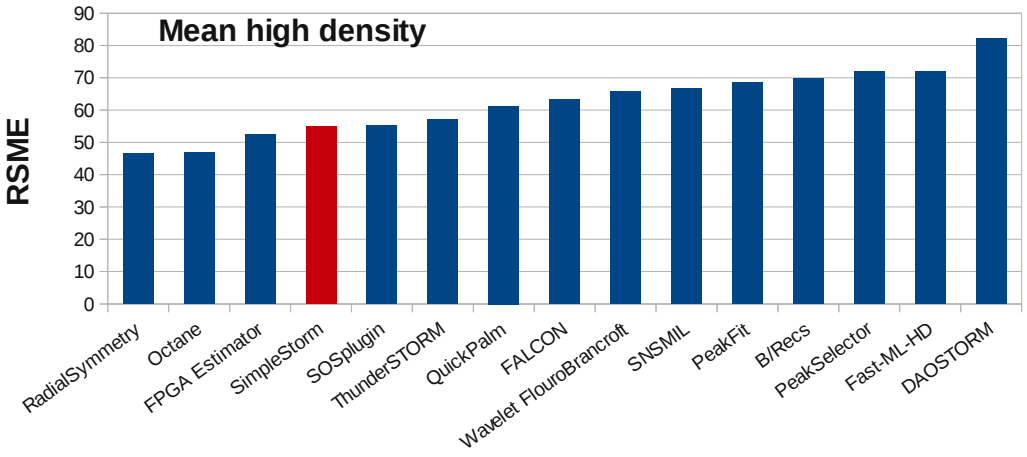


Figure 9.2: Results for high density data sets. Averaged RSME scores over all three datasets. For this score lower is better.

Table 9.4: Results for the main submission for low density data sets(with postprocessing)

Dataset	Jaccard (in %)	Precsision (in %)	Recall (in %)	RMSE (in nm)
LS1	86.12	100	86	26.32
LS2	69.64	97	71	37.43
LS3	47.48	99	48	54.2

Table 9.5: Results for the high precission submission for low density data sets

Dataset	Jaccard (in %)	Precsision (in %)	Recall (in %)	RMSE (in nm)
LS1	83.39	100	83	27.91
LS2	63.21	100	63	39.57
LS3	40.82	100	41	49.55

Table 9.6: Results for the high score submission for low density data sets (without postprocessing)

Dataset	Jaccard (in %)	Precsision (in %)	Recall (in %)	RMSE (in nm)
LS1	87.23	99	88	28.10
LS2	65.95	89	72	44.60
LS3	40.82	96	48	54.40

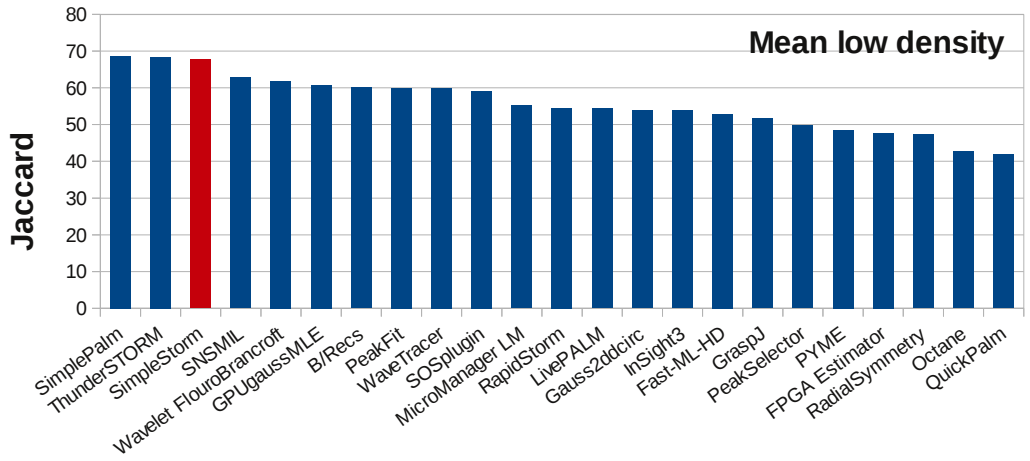


Figure 9.3: Results for low density data sets. The Jaccard indices are averaged over all three data sets. For this score higher is better.

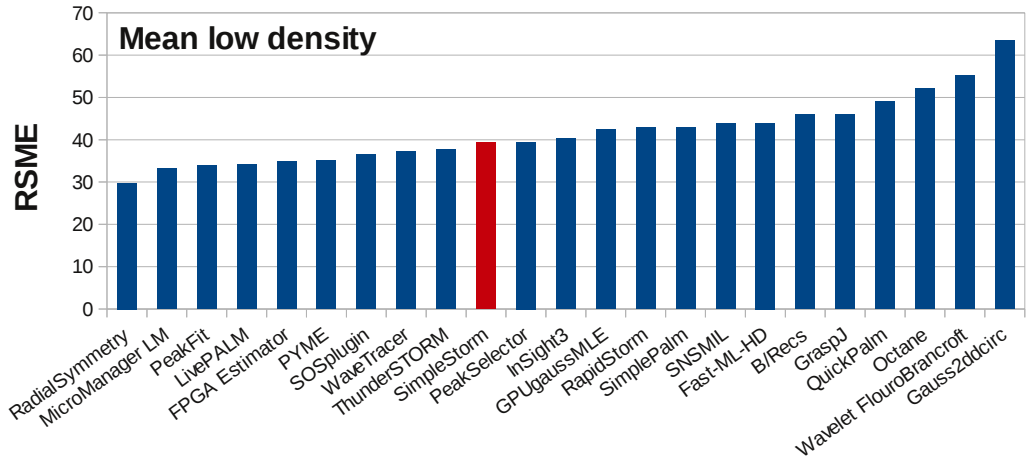


Figure 9.4: Results for low density data sets. Averaged RSME scores over all three datasets. For this score lower is better.

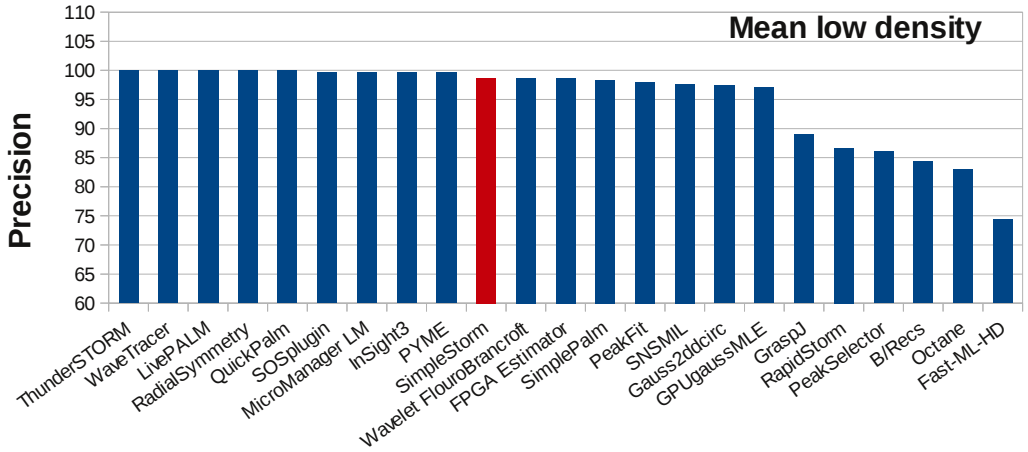


Figure 9.5: Results for low density data sets. Averaged Precision over all three datasets. For this score higher is better. The y axis is broken to show the differences better.

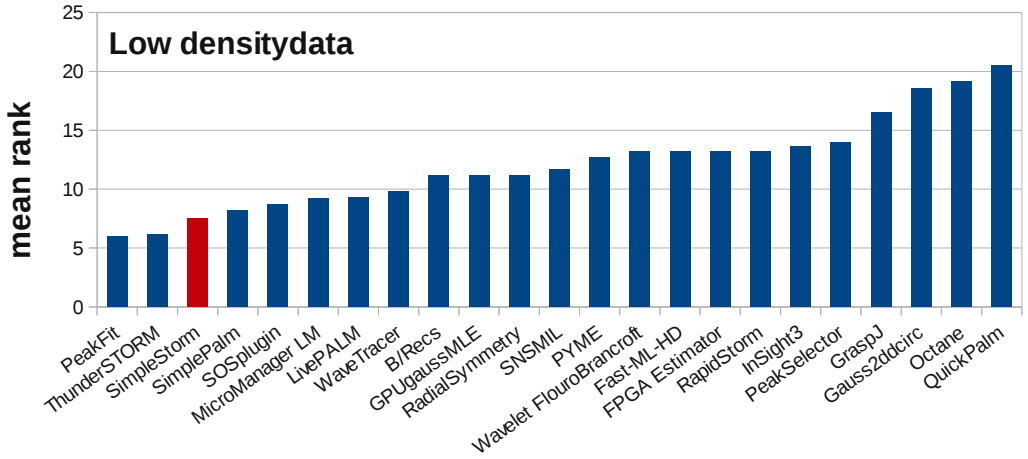


Figure 9.6: Averaged rank over Jaccard index and RSME score for all low density data sets. Lower ranks are better.

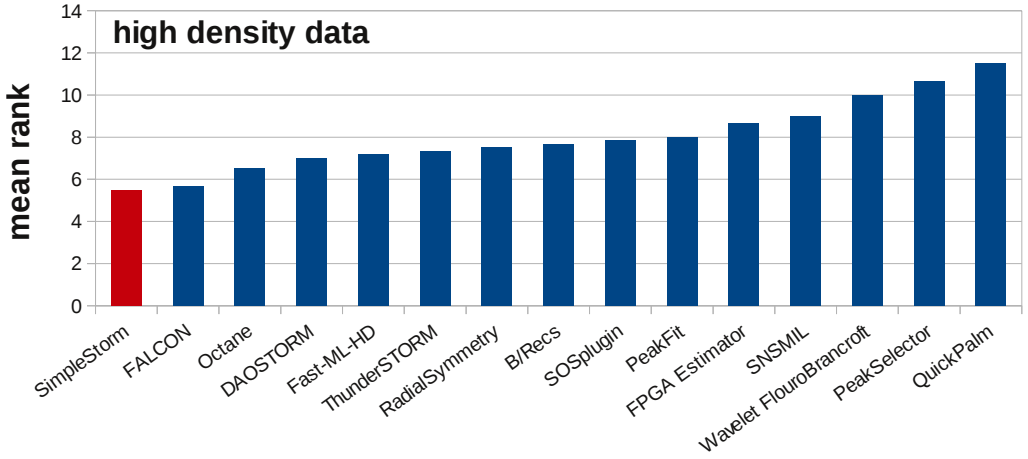


Figure 9.7: Averaged rank over Jaccard index and RSME score for all high density data sets. Lower ranks are better

10 Bibliography

Abbe, Ernst (1873), "Beiträge zur Theorie des Mikroskops und der mikroskopischen Wahrnehmung." *Archiv für Mikroskopische Anatomie*, 9, 456.

Alex D. Herbert, Anthony M. Carr, Thomas Etheridge (2013), "Peakfit - single-molecule localisation software for imagej." URL `http://www.sussex.ac.uk/gdsc/intranet/microscopy/imagej/smlm_plugins`. Accessed on 17.05.2013.

Anscombe, F. J. (1948), "The Transformation of Poisson, Binomial and Negative-Binomial Data." *Biometrika*, 35, 246–254, URL `http://www.jstor.org/stable/2332343`.

Baddeley, David, MarkB. Cannell, and Christian Soeller (2011), "Three-dimensional sub-100 nm super-resolution imaging of biological samples using a phase ramp in the objective pupil." *Nano Research*, 4, 589–598, URL `http://dx.doi.org/10.1007/s12274-011-0115-z`.

Betzig, E., G. H. Patterson, R. Sougrat, O. W. Lindwasser, S. Olenych, J. S. Bonifacino, M. W. Davidson, J. Lippincott-Schwartz, and H. F. Hess (2006), "Imaging intracellular fluorescent proteins at nanometer resolution." *Science*, 313, 1642–1645.

Biomedical Imaging Group, Switzerland, EPFL (2013), "Localization microscopy isbi 2013 challenge." URL `http://bigwww.epfl.ch/smlm/challenge/`. Accessed on 04.04.2013.

Carlas S Smith, Bernd Rieger, Nikolai Joseph and Keith A Lidke (2010), "Fast, single-molecule localization that achieves theoretically minimum uncertainty." URL `http://www.nature.com/nmeth/journal/v7/n5/full/nmeth.1449.html`.

E. M. M. Manders, J. A. Aten, F. J. Verbeek (1993), "Measurement of co-localization of objects in dual-colour confocal images journal of microscopy." *Journal of Microscopy*, 169, 375–382.

Fritschny, Jean-Marc and Wolfgang Härtig (2011), "Immunofluorescence." *eLS. John Wiley & Sons Ltd*, URL `http://www.els.net`.

Grüll, Frederik, Manfred Kirchgessner, Rainer Kaufmann, Michael Hausmann, and Udo Kebschull (2011), "Accelerating image analysis for localization microscopy with fp-gas." *In Field Programmable Logic and Applications*. Doi: 10.1109/FPL.2011.11.

Heilemann, Mike, Sebastian van de Linde, Mark Schüttpelz, Robert Kasper, Britta Seefeldt, Anindita Mukherjee, Philip Tinnefeld, and Markus Sauer

(2008), "Subdiffraction-resolution fluorescence imaging with conventional fluorescent probes." *Angewandte Chemie International Edition*, 47, 6172–6176, URL http://dx.doi.org/10.1002/anie.200802376.

Hell, S. W. and J. Wichmann (1994), "Breaking the diffraction resolution limit by stimulated emission: stimulated-emission-depletion fluorescence microscopy." *Optics Letters*, 19, 780–782, URL http://dx.doi.org/10.1364/ol.19.000780.

Jones, Eric, Travis Oliphant, Pearu Peterson, et al. (2001–), "SciPy: Open source scientific tools for Python." URL http://www.scipy.org/.

Köthe, Ullrich (2007), "Reliable low-level image analysis." URL http://hci.iwr.uni-heidelberg/Staff/ukoethe/papers/habil-koethe.pdf. Habilitation Tesis.

Köthe, Ullrich (2011), "The vigra computer vision library." URL http://hci.iwr.uni-heidelberg.de/vigra. Accessed on 31.04.2013.

Malkusch, Sebastian, Ulrike Endesfelder, J. Mondry, M. Gellé ri, P.J. Verveer, and Mike Heilemann (2011), "Coordinate-based colocalization analysis of single-molecule localization microscopy data." *Histochem Cell Biology*.

Newberry, Michael (1998), "Pixel response effects on ccd camera gain calibration." URL $http://www.mirametrics.com/tech_note_ccdgain.htm$. Tech note.

Rodgers, Joseph Lee and W. Alan Nicewander (Feb., 1988), "Thirteen ways to look at the correlation coefficient." *The American Statistician*, 42, pp. 59–66, URL http://www.jstor.org/stable/2685263.

Rousseeuw, Peter J. and Katrien van Driessen (1999), "A Fast Algorithm for the Minimum Covariance Determinant Estimator." *Technometrics*, 41, 212–223, URL http://www.jstor.org/stable/1270566.

Rust, X. Zhuang, M.; M. Bates (2006), "Sub-diffraction-limit imaging by stochastic optical reconstruction microscopy (storm)." *Nature Methods*, 3, 793–796.

Schleicher, Joachim (2011), *Image Processing for Super-Resolution Localization Microscopy Utilizing an FPGA Accelerator*. Master's thesis, Heidelberg University.

Seamus J Holden, Achillefs N Kapanidis, Stephan Uphoff (2011), "Daostorm: an algorithm for high- density super-resolution microscopy." URL http://www.nature.com/nmeth/journal/v8/n4/full/nmeth0411-279.html.

Starr, Rebecca, Shane Stahlheber, and Alex Small (2012), "Fast maximum likelihood algorithm for localization of fluorescent molecules: erratum." *Opt. Lett.*, 37, 1967–1967, URL http://ol.osa.org/abstract.cfm?URI=ol-37-11-1967.

WOLTER, S. and M. SAUER (2012), "Follow-up to paper by S. Wolter, M. Schüttpelz, M. Tscherepanow, S. van de Linde, M. Heilemann and M. Sauer, entitled Real-Time Computation of Subdiffraction-Resolution Fluorescence Images." *Journal of Microscopy*, 245, 109–109, URL http://dx.doi.org/10.1111/j.1365-2818.2011.03562.x.

Wolter, Steve, Ulrike Endesfelder, Sebastian van de Linde, Mike Heilemann, and Markus Sauer (2011), "Measuring localization performance of super-resolution algorithms on very active samples." *Opt. Express*, 19, 7020–7033, URL http://www.opticsexpress.org/abstract.cfm?URI=oe-19-8-7020.

Danksagung

Allen voran möchte Ulli für die gute Betreuung und die wertvollen algorithmischen Vorschläge und Ratschläge danken.
Vielen Dank an Leonie für die viele Unterstützung und den moralischen Beistand.
Buote war immer sehr hilfsbereit und geduldig, vielen Dank dafür.
Die Zusammenarbeit mit Benni war sehr harmonisch und gewinnbringend, vielen Dank für die Erklärungen zu STORM und für die gute Zusammenarbeit.
Die meisten Fortschritte bei der Gestaltung und der Implementierung der neuen Features wurden in der Zeit von Ilias Praktikum am HCI gemacht. Danke für die algorithmische Ideen und die gute Zusammenarbeit.
Luca möchte ich für seine Unterstützung danken.
Vielen Dank an Charlotte für das Korrekturlesen.
Zuletzt möchte ich noch Christel ganz herzlich für die Anmerkungen zu Aufbau und Inhalt danken.

Erklärung:

Ich versichere, dass ich diese Arbeit selbstständig verfasst habe und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe.

Heidelberg, den 8. Juni 2013 .....